NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

IMPROVING MARITIME DOMAIN AWARENESS USING NEURAL NETWORKS FOR TARGET OF INTEREST CLASSIFICATION

by

Brian M. Schaus

March 2015

Thesis Advisor: James Scrofani
Co-Advisor: Murali Tummala

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704–0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2015	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE
IMPROVING MARITIME DOMAIN AWARENESS USING NEURAL NETWORKS FOR TARGET OF INTEREST CLASSIFICATION

5. FUNDING NUMBERS

6. AUTHOR(S) Brian M. Schaus

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

Techniques for classifying maritime domain targets-of-interest within images are explored in this thesis. Geometric and photometric features within each image are extracted from processed images and are used to train a neural network. The trained neural network is tested with features of a known object. In the binary classification case, the neural network is used to determine whether a ship is present or not present in the image. In the multi-class and multi-level classification cases, the neural network is used to determine if the object belongs to one of four classes specified: warship, cargo ship, small boat, or other.

The Hough transformation is used to identify and characterize linear patterns exhibited by objects in images. As an alternative to geometric and photometric features to classify targets-of-interest, these linear patterns are used to train a neural network. The performance of the neural network is then tested for binary, multi-class, and multi-level classification schemes. The development of neural-network-based techniques for automated target-of-interest classification is a significant result of this thesis.

14. SUBJECT TERMS
Neural networks, feature extraction, Hough transformation, object detection, object classification, maritime domain awareness

15. NUMBER OF PAGES 101

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540–01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

IMPROVING MARITIME DOMAIN AWARENESS USING NEURAL NETWORKS FOR TARGET OF INTEREST CLASSIFICATION

Brian M. Schaus
Lieutenant, United States Navy
B.S., United States Naval Academy, 2008

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 2015

Author: Brian M. Schaus

Approved by: James Scrofani
Thesis Advisor

Murali Tummala
Co-Advisor

R. Clark Robertson
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Techniques for classifying maritime domain targets-of-interest within images are explored in this thesis. Geometric and photometric features within each image are extracted from processed images and are used to train a neural network. The trained neural network is tested with features of a known object. In the binary classification case, the neural network is used to determine whether a ship is present or not present in the image. In the multi-class and multi-level classification cases, the neural network is used to determine if the object belongs to one of four classes specified: warship, cargo ship, small boat, or other.

The Hough transformation is used to identify and characterize linear patterns exhibited by objects in images. As an alternative to geometric and photometric features to classify targets-of-interest, these linear patterns are used to train a neural network. The performance of the neural network is then tested for binary, multi-class, and multi-level classification schemes. The development of neural-network-based techniques for automated target-of-interest classification is a significant result of this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

A	area
C	complexity
MDA	maritime domain awareness
P	perimeter
RP	resilient backpropagation
S	spreading
SCG	scaled conjugate gradient

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The research detailed in this thesis is a development of a classification mechanism that identifies targets-of-interest from large volumes of maritime domain images. Essentially, the classification mechanism locates ship images within an image set of commonly found objects in the maritime domain and attempts to distinguish the class of ship that is desired. Target-of-interest classification reduces the number of images in the given large image set to a manageable size for an analyst to make the final determination.

In order to separate desired images from the rest, distinguishing attributes of objects must be selected carefully. The images collected of the maritime domain contain objects of various sizes and shapes; however, certain object classes bear similar geometric features to other object classes. In this thesis, we exploited the geometric and photometric features as a method of distinguishing between images that contain a target-of-interest and images that have no value.

An additional set of features is extracted using the Hough transformation. The Hough transformation is a commonly used image processing technique for detecting linear and parametric shapes. By performing the Hough transformation of the maritime domain images, we can extract a valuable set of linear and spatial features of objects within the images.

The neural network as a viable machine learning tool is explored in this work. Neural networks are trained using the geometric, photometric, and Hough transformation features to classify targets-of-interest in a large set of images. Neural network topologies are studied by varying the size of the hidden layer. Two learning algorithms were explored for training the neural networks: resilient backpropagation (RP) and scaled conjugate gradient backpropagation (SCG). The results of the neural network training performance are presented using mean squared error convergence plots. In all implementations, the SCG learning algorithm converges the quickest.

Three types of classification are explored. In binary classification, ship images are separated from the other images. The results of classification are presented as

performance plots. Based on 1000 trials using geometric and photometric features, the SCG learning algorithm and ten hidden neurons, ship images were correctly classified at a rate of 94.7 percent.

In multi-class classification, separate neural networks are trained using the same extracted features and then tested to classify multiple classes of ships. These neural networks performed multi-class classification by attempting to distinguish between warship, cargo ship, small boat, or non-ship images. Based on 1000 trials using geometric and photometric features, the SCG learning algorithm and ten hidden neurons, warship images were correctly classified at a rate of 62.5 percent.

A multi-level approach to multi-class classification is developed. The goal of the multi-level approach is to have only warship images remain within the image set by performing a process of elimination. The approach accomplished this by implementing a series of binary classifications to eliminate one or more possible object classes from the image set. The first binary classification separates ship images from other images. The second binary classification separates big ship images from small ship images. The final binary classification separates warship images from cargo ship images. The results of each binary classification are presented as performance plots.

The significant contribution in this thesis is the exploration of neural-network based learning and classification techniques to classify targets-of-interest in the ever-expanding maritime domain in an automated manner. The separation of images of value from other images using neural networks will enable a user to allocate more time for further analysis on images of interest. The feasibility to train neural networks to classify ship from non-ship images was demonstrated in this thesis; however, further exploration of multi-class and multi-level classification is recommended.

ACKNOWLEDGMENTS

I cannot express enough thanks to my advisors for their continued support and encouragement: Dr. James Scrofani, my thesis advisor, and Dr. Murali Tummala, my co-advisor. I offer my sincere appreciation for the learning opportunities provided and continued support throughout the duration of my research and thesis processing.

My completion of this thesis could not have been accomplished without the tremendous support of the Naval Postgraduate School's Center for Multi-INT Studies. Special thanks to Mrs. Deborah Shifflett for providing an office and resources for writing my thesis, and assisting in the travel plans to visit my thesis sponsor.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The National Security Presidential Directive (NSPD-41)/Homeland Security Presidential Directive-13 in support of the Military Security Policy (MSP) was issued by President George W. Bush on December 21, 2004 in an effort to enhance homeland security and protect U.S. maritime interests [1]. In this directive, the president stresses the need for increasing Maritime Domain Awareness (MDA). MDA encompasses anything associated with the global maritime domain that could impact the United States [1]. The efforts made to increase MDA are critical to sustaining global economic stability and growth and are vital to the interests of the United States.

Due to its complexity and size, the maritime domain is susceptible to threats and unlawful activity [2]. It is important to identify threats as early as possible by integrating intelligence, surveillance, observation, and navigation systems into a common operating picture [1]. As the exploitation of the maritime domain increases and as the common operating picture continues to grow with information, the need for time-sensitive decision making becomes imperative. With the increase of information collection, it is impossible for a human operator to analyze all of the data accumulated; therefore, there is a necessity for automated technology to process data and provide viable intelligence going forward.

The research detailed in this thesis is a development of techniques to detect targets-of-interest from large volumes of maritime vessel images. Implementing these techniques will directly assist the effort to increase MDA.

A. THESIS OBJECTIVE

The objective of this thesis is to contribute to MDA by developing an automated detection mechanism that identifies targets-of-interest from large volumes of maritime domain images. Target-of-interest detection reduces the number of images in the image set to a manageable size for an analyst. In order to separate desired images from the rest, distinguishing attributes of objects must be selected carefully.

Commonly found objects in the maritime domain encompass infinite variation in size and shape; however, certain object classes bear similar geometric and photometric

features to other object classes. In this thesis, we exploit these geometric and photometric features as a method of distinguishing between images that contain a target-of-interest and images that have no value.

The Hough transformation is used to identify linear patterns and parametric shapes in images and is robust against noise and occlusion [3]. By performing the Hough transformation against the maritime domain images, we extract a valuable set of linear and spatial features of objects within the images. As an alternative method to geometric and photometric feature extraction, we use the Hough transformation features for target-of-interest detection.

The exploration of neural networks as a viable machine learning application is researched. Neural networks are trained using the geometric, photometric, and Hough transformation features extracted to first distinguish whether a ship is present or not present in the image. The test is a binary classification, which provides only a positive or negative response that a ship is present. For further exploration, separate neural networks are trained using the same extracted features and then tested to classify multiple classes of ships. These neural networks perform multi-class classification by attempting to distinguish between warship, cargo ship, small boat, or non-ship images. Using the combination of binary and multi-class classifications, we determine whether a ship is present in the image, and, if a positive classification is obtained, we then determine which class of ship is present.

B. RELATED WORK

Neural networks have received increasing interest because of their ability to learn from patterns that humans cannot easily identify and to use their learning to classify the objects of interest. In this thesis, we explore relevant work on neural networks that fall into two categories: identifying shapes in images by training neural networks with (i) extracted geometric and photometric features and (ii) the application of the Hough transformation.

Neural networks were investigated in [4] for the potential to identify oil spills in the ocean using European remote sensing-synthetic aperture radar data. The input to the

neural network consists of a set of features that depict an oil spill candidate, and the output provides the probability that the candidate is an actual oil spill [4]. The features collected from the images used reflected the geometry of oil spill candidates in terms of their extension and shape, as well as the physical behavior of backscattering pixel intensity of the object, the background, and the area around the border of the object [4]. The features extracted include area, perimeter, complexity, spreading, object standard deviation, background standard deviation, maximum contrast, mean contrast, maximum gradient, mean gradient, and gradient standard deviation. From the training of the neural network using these features, oil spills were successfully detected with a high probability. Although [4] does not research the potential to use these features for detecting ships and other maritime domain objects, the techniques used in [4] can be extended for the purposes of this thesis.

The Hough transformation was explored in [3] for two-dimensional shape recognition using a neural network. The inputs to the neural network were a combination of two complimentary features extracted from the Hough space that allowed for recognition of general shapes [3]. Two-dimensional shapes were successfully recognized using these features to train a neural network. The shape recognition technique proposed in [3] is applied to target-of-interest detection in this thesis.

C. ORGANIZATION

Five chapters and four appendices are contained in this thesis. The background information related to neural networks and the Hough transformation is covered in Chapter II. Image preparation techniques, feature extraction techniques, and proposed schemes for classification using neural networks are described in Chapter III. Specific details about how the techniques from Chapter III are employed in this research are included in Chapter IV. A summary of key results and considerations for follow-on work is provided in Chapter V. The MATLAB code for image preparation is contained in Appendix A. The MATLAB code for feature extraction is contained in Appendix B. The MATLAB code for the Hough transformation and feature extraction are contained in

Appendix C. The MATLAB code for training, testing, and classification using MATLAB'S Neural Network Toolbox is contained in Appendix D.

II. BACKGROUND

Three concepts that are utilized for target-of-interest classification are explored in this chapter. First, image preparation and feature extraction are discussed. Next, the Hough transformation operation, and how it is applied to images to extract valuable features, is detailed. Finally, the structure and functionality of neural networks are presented.

A. IMAGE PREPARATION

The object classification methods used in this thesis rely heavily on extracting features that best represent the objects in the images. Raw images alone do not provide neural networks with sufficient information during the training phase to accurately output a desired response [5]. In order to correct for this, the raw images in the collection require processing before extracting features. Just as humans use features of a face, body, and body movements to recognize a particular person, neural networks use features to learn and associate to a particular entity. In the next subsection in this chapter, feature extraction is described and the features that are used for the applications in this thesis are covered.

1. Feature Extraction

Two types of features are extracted from the images in this technique: geometric features and photometric features. The geometric features take advantage of the object's size, shape, and spreading throughout the image [4]. The photometric features take advantage of the textual variations within the object's region by analyzing the difference in intensity pixel values of the object, background, and edges around the border of the object [4]. The description of these features and how they are calculated are further explained in the remainder of this section.

a. Area

The area (A) feature is defined as the area of the object within the image. This is calculated by summing the number of white pixels in the segmented image [4]. The image in Figure 1 illustrates the region that determines area.

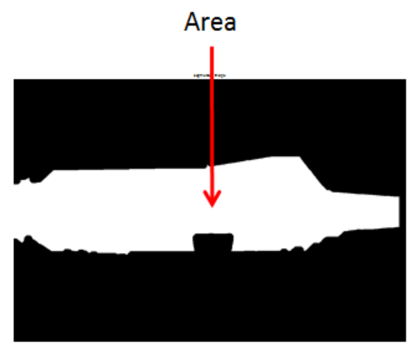


Figure 1. A segmented object illustrating region for area. The area is calculated by summing the white pixels within the borders of the object.

b. Perimeter

The perimeter (P) feature is defined as the length of the object's border. This is calculated by counting the number of white pixels along the border of the segmented object [4]. The image in Figure 2 illustrates the region that determines perimeter.

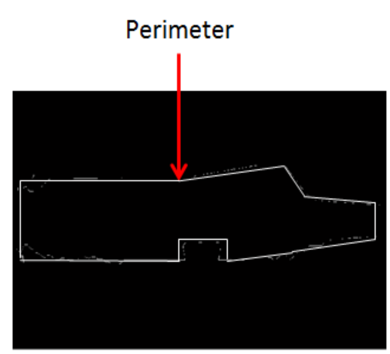


Figure 2. A segmented object illustrating the region for perimeter. The perimeter is calculated by summing the white pixels bordering the object.

c. Complexity

The complexity (C) feature is defined as a function of the area and perimeter of the object in the image. This feature measures the obscureness of the object's geometry. In general, small values of C represent simple geometry [4]. Conversely, larger values of C represent a complex geometric shape. Complexity is calculated as

$$C = \frac{P}{2\sqrt{\pi A}}, \tag{1}$$

where P is the perimeter of the object and A is the area of the object [4].

d. Spreading

The spreading (S) feature is defined as level of dispersion the object has in the image [4]. In general, large values of S represent objects of a circular shape, whereas small values represent objects that are long and narrow [4]. Spreading is calculated as

$$S = \frac{100\lambda_1}{\lambda_1 + \lambda_2}, \tag{2}$$

where λ_1 and λ_2 are the eigenvalues associated with the covariance matrix of the processed image and $\lambda_1 > \lambda_2$ [4].

e. Object Standard Deviation

The object standard deviation (σ_{obj}) feature is defined as the standard deviation (in dB) of pixel intensity values I_o belonging to the object. This is calculated by computing the standard deviation of the object's intensity values in dB as given by

$$\sigma_{obj}(dB) = 10\log\left(\frac{1}{\sigma_{obj}}\right), \tag{3}$$

where σ_{obj} is the standard deviation of the object's intensity values [4].

f. Background Standard Deviation

The background standard deviation (σ_{back}) feature is defined as the standard deviation (in dB) of pixel intensity values I_B belonging to the background. This is calculated by computing the standard deviation of background intensity values in dB using Equation 3 [4].

g. Contrast

Contrast is defined as the difference in intensity values of I_o and I_B. The maximum contrast (κ_{max}) feature is calculated by the difference (in dB) between the average background intensity value and the lowest intensity value of the object [4]. The average contrast (κ_{av}) feature is calculated by the difference (in dB) between the mean background intensity value and the mean object intensity value [4].

h. Gradient

The border gradient (δ) is defined as the change in intensity values from pixel to pixel along the object's border and is calculated as

$$\delta = \frac{dI}{dx}, \tag{4}$$

where I is the pixel intensity value and x is the an individual pixel. The maximum gradient (δ_{max}) feature is defined as the maximum value (in dB) of the border gradient [4]. The average gradient (δ_{av}) feature is defined as the average border gradient represented in dB [4]. The gradient standard deviation (σ_{grad}) feature is defined as the standard deviation of the border gradient values in dB [4].

2. Feature Selection

It is important to select features that are relevant to the objects to be classified. The features must accurately depict information from image data that support the neural network in the learning phase [6]. The neural network is unable to determine if it is trained with poorly selected features. The number of features selected to train a neural network must also be selective. There is a tipping point at which the addition of features leads to poorer classification performance [6]. If there is redundant input information representing certain features, the neural network undergoes a longer training phase. Redundant input information forces the neural network to place lower emphasis on these features, when in actuality the features are beneficial to classification performance. There must be a balance between selecting features with high information value while limiting the number of input values to the neural network. Carefully choosing features leads to faster training times and better classification results.

The goal of feature selection is to retrieve the features with strong relevance. A feature with strong relevance implies that classification performance degrades significantly with the removal of that feature, whereas weak relevance implies that a feature has little to no significance to the classification performance [6]. The relevance of each of the features extracted is examined in Chapter IV Section B.3.

B. THE HOUGH TRANSFORMATION

The human eye is capable of discerning lines and shapes of images, while a computer requires additional computations to detect those lines and recognize different shapes. The Hough transformation is a computational tool for such line and spatial detection. The techniques proposed in this thesis use the Hough transformation methods outlined in [3] and tailor those methods to accommodate maritime domain images.

1. Common Uses of the Hough Transformation

The Hough transformation is commonly used to detect linear patterns in images but can be adapted as an effective method of detecting parametric shapes as well [3]. Features extracted using the Hough transformation have proven to be successful in training neural networks to distinguish image patterns. An example of the Hough transformation working in conjunction with neural networks is investigated in [3] for the recognition of alphabetic letters, as well as complex images with convexity and concavity. Another example of the Hough transformation working together with neural networks is developed in [7] for autonomous mobile robots using feature-based localization to detect walls in an unknown environment. Another application of the Hough transformation is explored in [8] for seismic pattern detection. The robustness of the Hough transformation has proven useful in pattern recognition applications and is explored for the target-of-interest classification in this thesis.

2. Hough Transformation Technique

An image can be represented on an (x, y) coordinate plane in which each pixel has an associated (x, y) coordinate. The Hough transformation takes each pixel in the (x, y) plane and translates them to the Hough space plane. The Hough space is represented by a (θ, r) coordinate plane. A straight line in the (x, y) coordinate system can be represented by a single point in the (θ, r) coordinate plane using the relationship

$$r = x\cos(\theta) + y\sin(\theta)\,, \tag{5}$$

where r is the shortest distance from the image's origin to the line, and θ is the angle that is formed about the x-axis from the line's normal to the origin [7]. A single point in the (x, y) plane can be mapped to a sinusoidal curve in the (θ, r) plane. All of the points along the sinusoidal curve represent the infinite combinations of θ and r values that satisfy Equation 6 [7]. These concepts are illustrated by the plots in Figure 3.

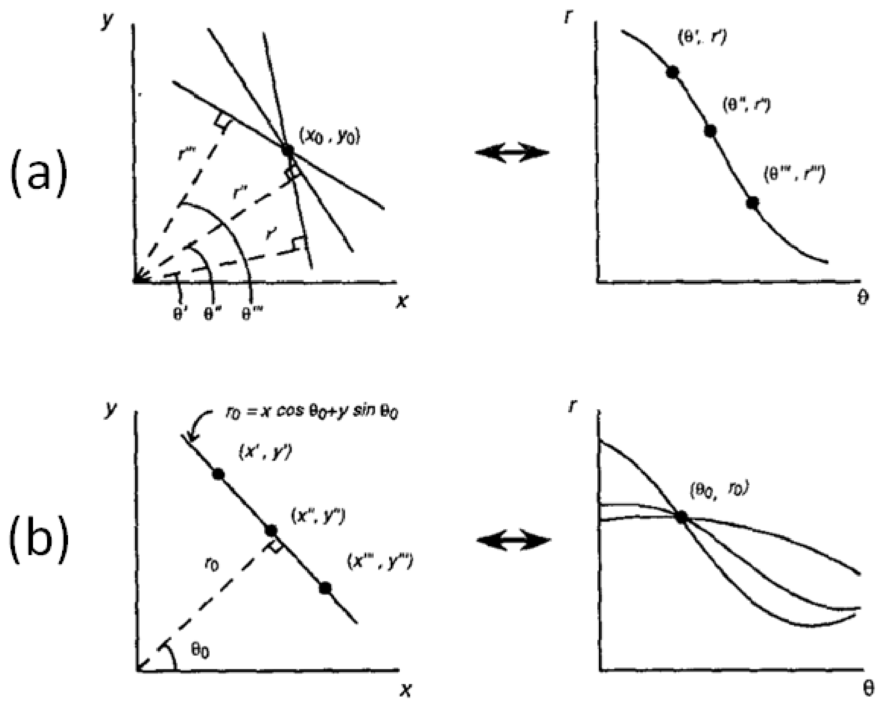


Figure 3. The point-curve transformation converts pixels from the (x, y) coordinate plane to the (θ, r) Hough space. (a) A point in the (x, y) coordinate plane is mapped to a sinusoidal curve in the (θ, r) plane. (b) Collinear points transform into curves intersecting at a single point in the (θ, r) plane. From [7].

It is shown in Figure 3 that each point along the line in the (x, y) plane corresponds to a sinusoidal curve in the (θ, r) plane. The intersection of the curves are the (θ, r) values for a common line in the (x, y) plane [7]. The Hough transformation uses the intersection points in the (θ, r) plane to detect lines. The number of intersections for each (θ, r) component is summed and entered in the Hough space. Large values in the Hough space translate to a strong detection of a line in the (x, y) plane.

Before the Hough transformation can be applied to an image, the image must be converted into grayscale which represents the pixel intensities on a scale from 0 to 255 for 8-bit resolution. The presence of a line in a grayscale image is detected by the string of pixel values of equal intensity. Pixels with equal intensities are translated into the Hough space. When using images with noise and weak line correlation, such as the images used in this thesis, the grayscale images require an additional image processing technique, known as edge detection. This step removes the noise and weak line correlation and only returns lines for which the changes in the intensity values from pixel to pixel are a maximum. Generally, the maximum changes in intensity values correspond only to the object in the image. The edge detection offers a cleaner representation of the object of interest, which in turn yields more accurate feature values.

C. NEURAL NETWORKS

Neural networks were developed to mimic the human brain. The center of the human nervous system is the brain, which continually receives information from all senses, characterizes that information, and makes proper decisions [9]. The brain is able to make these decisions from the interaction between electrical signals that carry the information and the neurons surrounding the brain. There are an estimated 10 billion neurons in the brain, with 60 trillion connections between them [9]. These neurons adapt and learn from the electrical impulses they receive, and communicate their knowledge with the other neurons of the brain [9]. From the positive and negative responses that modify the neurons in the brain, humans are capable of distinguishing between patterns. Neural networks train and perform in a manner similar to the human brain. Neural networks are trained from incoming information to output a desired response.

Neural networks were chosen as a viable solution for MDA because of their ability to learn and adapt from the training they undergo. There are a vast number of vessels within the maritime domain, which makes it difficult for a human operator to locate and distinguish each one by eye. Neural networks used in this application because of their ability to generalize and produce reasonable solutions for inputs that were not encountered in the training phase [9]. It is impossible to train a neural network with every possible object found in the maritime domain, but their ability to classify objects of different sizes and shapes through unsupervised learning is tested in this thesis.

1. Common Uses of Neural Networks

Neural networks are most commonly used to analyze large amounts of data and detect trends in that humans may not necessarily detect. The neural networks then use these learned trends to make decisions. Although neural networks are used for numerous applications, they have been widely used for image recognition [3], [4], [7], [8].

2. Neural Network Topology

In order to comprehend how neural networks operate, there must be an understanding of how they are structured. The specifications of how inputs, outputs and hidden layers are connected within a neural network structure refer to its topology [10]. The objective of a network's topology is to maintain invariance by structure, which means the network structure is resistant to changes in classification rate when it is trained with modified versions of the same input [9]. The following is an introduction to the composition of a neural network and the types of neural network architecture that are explored in this thesis. In this section, we refer to the variables and their designations as given in Table 1.

Table 1. Variables are used to designate neural network values, vectors, and functions.

x	Feature values
X	Input vector
y	Output values
Y	Output vector
w	Weights
z	Output from node
φ	Activation function
ω	Output from activation function

A fully connected, multilayer neural network is investigated in this thesis. It is characterized by the presence of input, hidden, and output layers. Fully connected neural networks are defined as a topology in which every node in each layer is connected to every node of the adjacent layers [9]. The example neural network topology in Figure 4 illustrates how the input, hidden, and output layers communicate in a fully connected, multilayer neural network to produce a binary classification output. The input layer consists of source nodes, which carry the input data $(x_0, x_1, \ldots, x_{n-1})$. The number of source nodes (n) equals the number of input values applied to the neural network. The source nodes pass the values to the hidden layer, which contains computational nodes, called hidden neurons. The hidden neurons communicate effectively between the input layer and the output layer. There are no set standards for the number of hidden layers and hidden neurons within this layer. Multiple trials must be performed to determine the hidden layer structure that produces the best classification results. The outputs of the last hidden layer provide a response to the output nodes based on the information provided from the input layer. In binary classification, the output layer contains two output nodes to represent the two possible outcomes (y_0 and y_1).

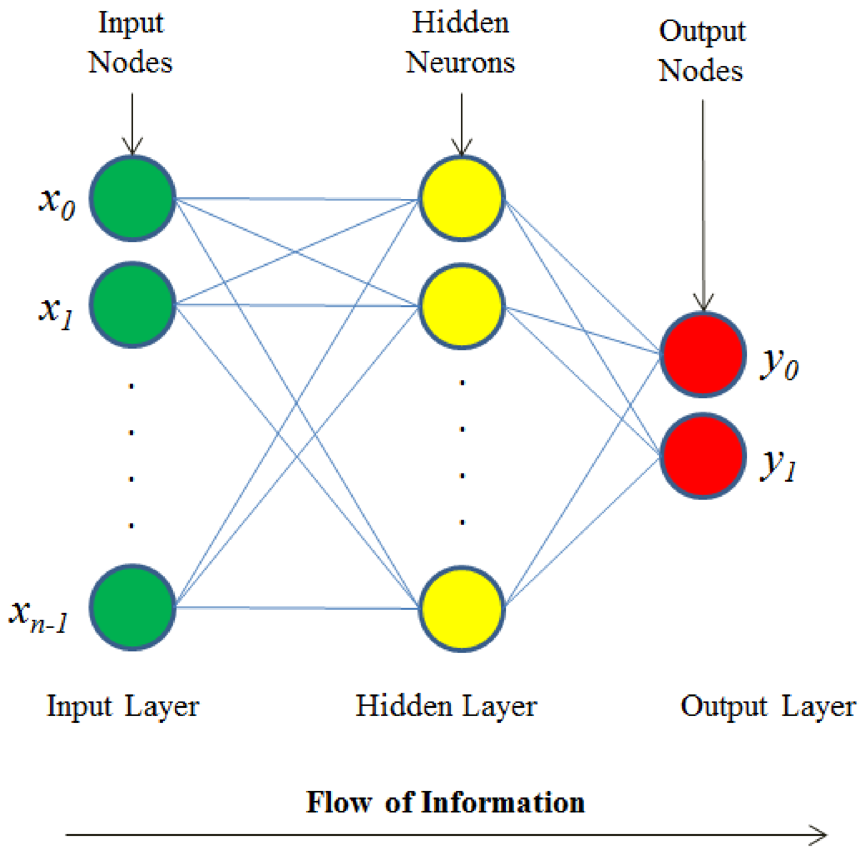


Figure 4. A neural network topology designed for binary classification has *n*-1 input nodes, variable number of hidden neurons, and two output nodes.

A fully connected, multilayer neural network is also designed to have a multi-class classification output. The topology of the input and hidden layers are assembled as they were in the binary classification network; however, the output layer contains output nodes $(y_0, y_1, \ldots, y_{m-1})$, where m is the number of possible classifications. The example neural network topology in Figure 5 illustrates how the input, hidden, and output layers communicate in a fully connected, multilayer neural network to produce multi-class classifications.

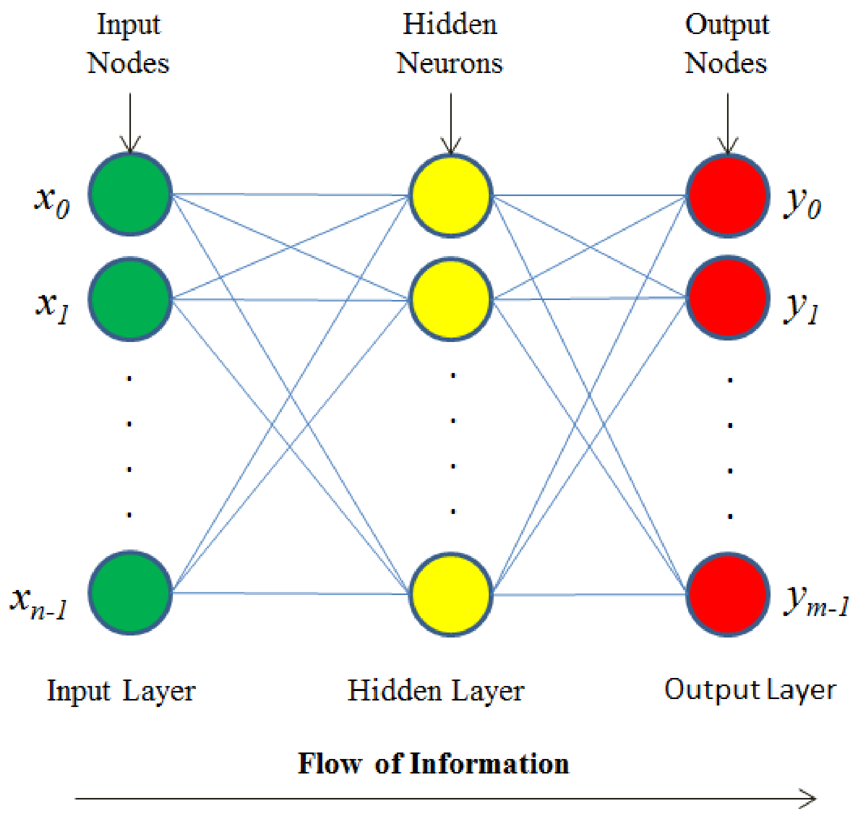


Figure 5. A neural network topology designed for multi-class classifications will have n-1 input nodes, variable number of hidden neurons, and m-1 output nodes.

3. Neural Network Training Process

The most valuable facet of a neural network is its ability to learn from the information that it is presented. Neural networks learn by adjusting the weights of the hidden neurons to produce an output that matches its target value. This method of learning is known as supervised learning [9]. Supervised learning is described as a closed-loop feedback system that adjusts for error correction. When hidden neurons are presented information from the input layer, their weights are modified to minimize the difference between the desired response and the actual response. Neural networks may use mean squared error as a measure of performance to fine-tune the neuron weights [9]. These weights are adjusted each iteration until the desired performance is achieved.

The building blocks of a neural network, and where the majority of the complex computations occur for adjusting weights, are the individual hidden neurons. An example

signal-flow graph of a hidden neuron is illustrated in Figure 6. The input vector X is normalized in the input layer and is multiplied by random weight values $(w_0, w_1, \dots, w_{n-1})$. The product of the normalized inputs and weights are then summed together in the hidden neuron. The sum of these products z is passed through an activation transfer function φ.

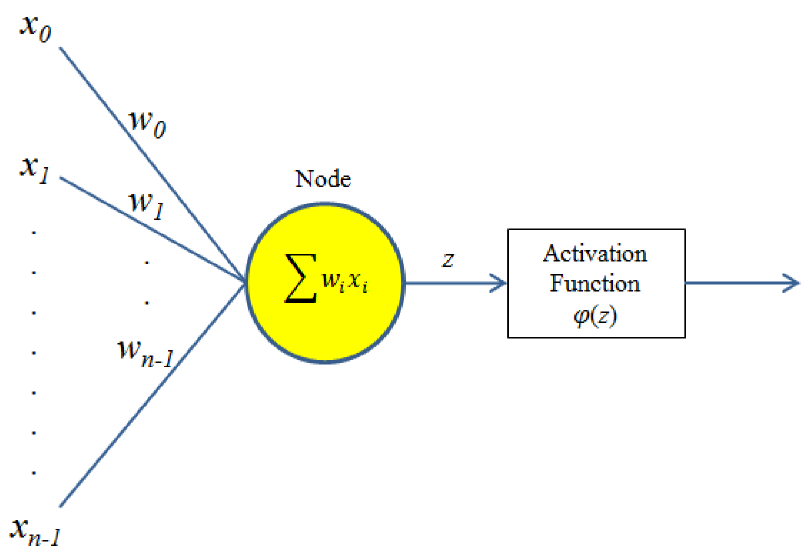


Figure 6. A hidden layer node is the junction where the product of the normalized input values and their weights are summed together. The sum z is passed through an activation function φ.

Activation functions may vary from neural network to neural network, but all of them act as the decision maker [9]. The activation function used for the neural network in this thesis is the logarithmic-sigmoid transfer function. Sigmoid is the most common form of activation function and is often used for pattern recognition problems [9]. The output of the sigmoid function is given by

$$\varphi(z) = \frac{1}{1+e^{-z}}, \tag{6}$$

where φ is the activation function and z is the sum of the weighted inputs. Equation (6) yields an output between zero and one.

During the training phase, there is an error-correction process to adjust the weights of the neural network. The error-correction process uses the gradient of $\varphi(z)$ and the error e to adjust the weights. The "S" curve of the sigmoid function is illustrated in Figure 7. The error-correction term ∇_e for adjusting the weight values is computed as [9]

$$\nabla_e = \frac{d\phi(z)}{dz} e, \tag{7}$$

where the error $e = d_\omega - \varphi(z)$, and d_ω is the desired output. This process is repeated until the slope of the curve is zero, which indicates there are no further corrections to be made to the input weights [9]. The signal-flow graph illustrating error-correction learning is depicted in Figure 8.

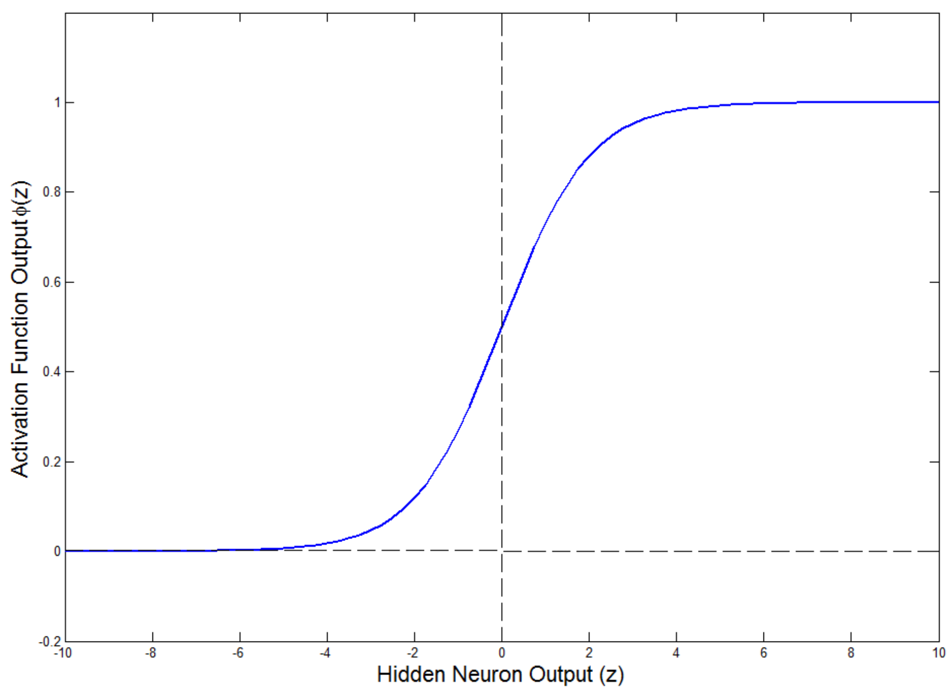


Figure 7. The "S" shape curve is a characteristic of the Logarithmic-Sigmoid Transfer Function. The output value of the activation function is determined from the summation of the product of normalized input values and their weights.

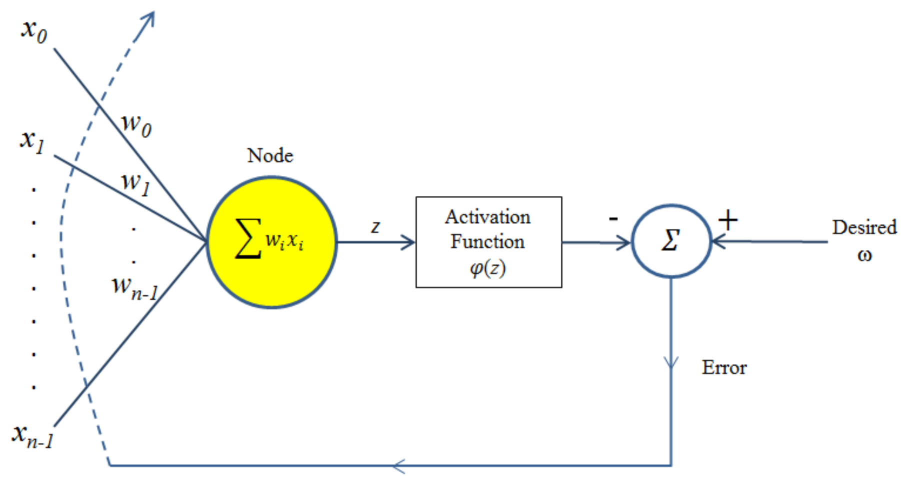


Figure 8. Error-correction is back-propagated to adjust the weights of the normalized input values.

The manner in which the neural network adjusts the weights for error correction depends on the learning algorithm that is used to train the neural network. A learning algorithm is a tool for adjusting weights, so the activation function convergences to zero slope [9]. Learning algorithms vary in the manner the weights are adjusted, but all aim towards optimized classification. Just as there are no set rules for selecting the number of hidden layers or hidden neurons when developing a neural network topology, there are no guidelines for selecting a learning algorithm [9]. Each algorithm has its advantages, but the advantages go unobserved until simulations have been performed.

Two learning algorithms are investigated in this thesis: resilient backpropagation (RP) and scaled conjugate gradient backpropagation (SCG). Training algorithms that use the magnitude of the slope often cause longer training times for neural networks since the magnitudes are small at the tails of the sigmoid function. The RP uses an update value α rather than the magnitude of the slope [11]. The update value is increased by a pre-determined value when the slope has the same sign for two consecutive iterations [11]. Conversely, the update value is decreased when the sign of the slope changes from the previous iteration [11].

The basic equation for the RP algorithm is given by

$$w_{k+1} = w_k + \alpha g(\nabla_e) \tag{8}$$

where the function $g(\cdot)$ tracks the sign changes in the gradient.

The SCG algorithm updates weights in a more complex fashion. This algorithm uses a combination of the Levenberg-Marquardt algorithm and the conjugate gradient approach [12]. The general equation for the SCG algorithm is given by [12]

$$w_{k+1} = w_k + \mu_k p_k \tag{9}$$

where μ_k is a variable step size and p_k is the search direction, which can be recursively computed from

$$p_{k+1} = \beta_k p_k - \nabla_{e_k} \tag{10}$$

and

$$\beta_k = \frac{\left|\nabla_{e_{k+1}}\right|^2 - \nabla_{e_{k+1}}^T \nabla_{e_k}}{p_k^T \nabla_{e_k}}\ . \tag{11}$$

4. Classification

The final stage in a neural network is classification, which assesses if the neural network was trained appropriately. This is performed by passing a set of inputs through a trained neural network. The output values of the activation function closely resemble one of the target values for a class. The target values that are the closest to the output determine the classification. The classification process is the same for binary and multi-class classifications.

Background concepts necessary for the development of techniques presented for locating and identifying targets-of-interest in the maritime domain were examined in this chapter. The necessity to prepare images, the methods to extract features, and the importance of feature selection were discussed. Additionally, the Hough transformation operation and its ability to recognize linear characteristics in images were described. Lastly, neural network topology, training, and classification were presented.

THIS PAGE INTENTIONALLY LEFT BLANK

III. AN IMAGE CLASSIFICATION SCHEME FOR MARITIME DOMAIN AWARENESS

One approach to improving MDA is to develop a process capable of separating images of significance from images of no importance. By separating the significant images from the others, a user may identify targets-of-interest more quickly and more efficiently. One such process following the object classification scheme in Figure 9 is developed in this chapter. Given the concepts outlined in Chapter II, this process encompasses all the steps taken to successfully identify targets-of-interest within an image set. The object classification scheme is applied to the geometric, photometric, and Hough transformation features. The chapter is divided into four parts. Image preparation for geometric and photometric feature extraction is explained in the first part; image preparation for Hough transformation feature extraction is covered in the second part; neural network training, testing, and classification are explained in the third part; and feature selection is covered in the fourth.

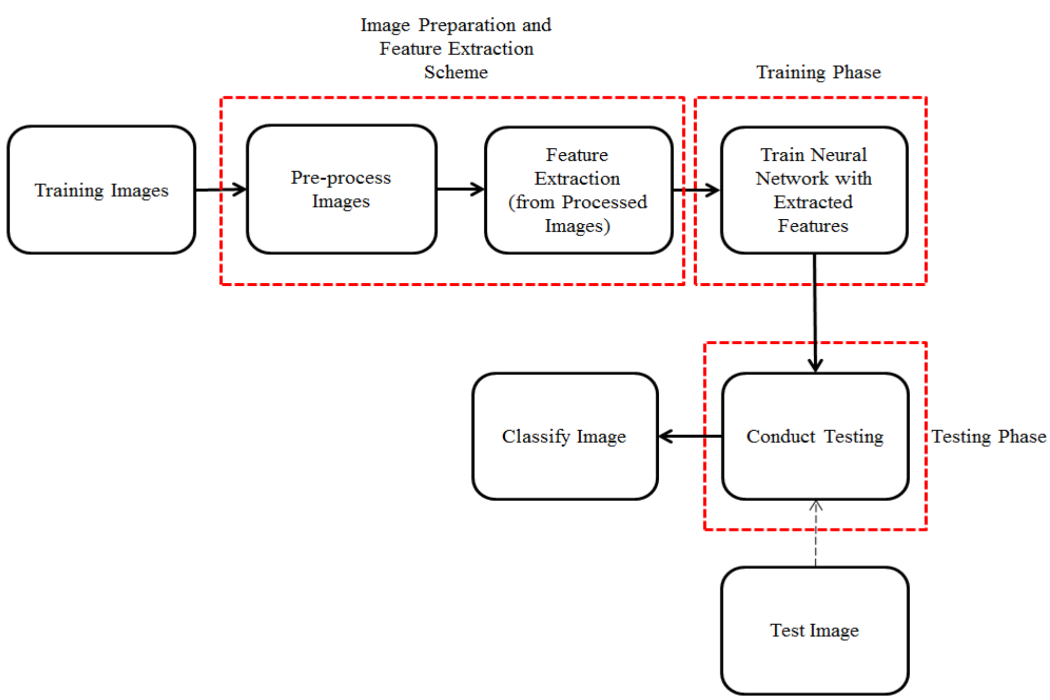


Figure 9. The object classification scheme encompasses the steps to prepare images, extract features, as well as train and test the neural network.

A. IMAGE PREPARATION FOR GEOMETRIC AND PHOTOMETRIC FEATURE EXTRACTION

The first step in the classification scheme is to obtain a training set of images. The more images collected, the better the neural network trains and the better the classification results are. There is a large amount of variation in an object's appearance from image to image [6]. Using a substantially sized image training set captures most of these variations and help the neural network in the training phase. The image training set must contain only suitable images. A suitable image in terms of this thesis is an 8-bit image of one maritime domain object from overhead or near overhead. This aspect was chosen to keep the images consistent and for ease of object segmentation while processing. Only one object must be present per image to ensure the features extracted are consistent with that of one target-of-interest.

Images from the training set are preprocessed before geometric and photometric features are extracted. Raw images do not carry information that is useful for training the neural network. Image preprocessing must take place in order to extract informative features. The image processing and geometric and photometric feature extraction scheme is illustrated in Figure 10.

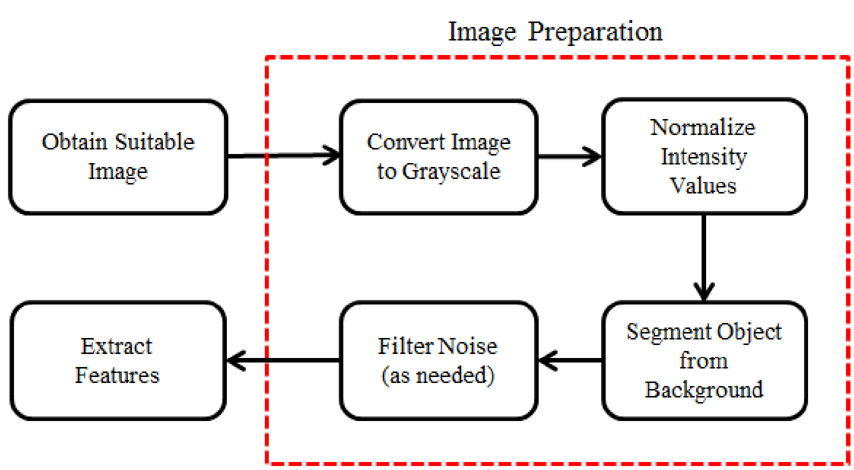


Figure 10. The image preparation scheme encompasses steps to segment the object from its background in order to extract geometric and photometric features for training.

The image preparation scheme begins by converting the images to grayscale. This step is performed to obtain the intensity information. The pixel values are in the range [0, 255] for 8-bit resolution. This step yields the image exclusively in shades of gray, which varies from black at the weakest intensity to white at the strongest. After the images are transformed to grayscale, the pixel intensity values are normalized so that their values range from 0 to 1. This is obtained by dividing all of the pixel values by 255. The next step in the image preparation scheme is to segment the objects in the images from the background. This is accomplished by first plotting a histogram of the normalized pixel intensity values. A typical histogram of normalized pixel intensity values is provided in Figure 11.

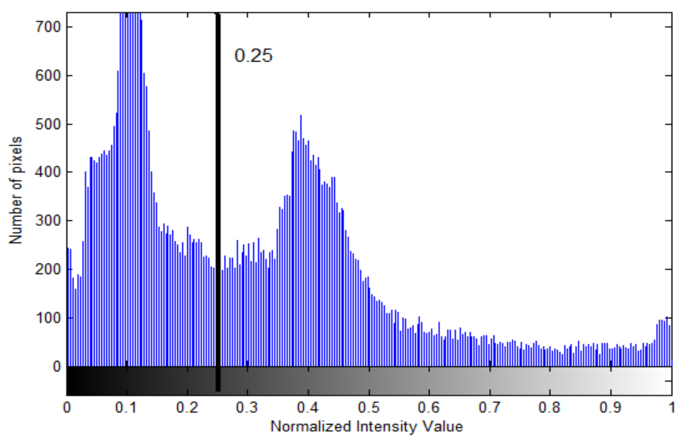


Figure 11. A histogram of the normalized pixel intensity value. The solid line corresponds to the local minimum between the two peaks and indicates the threshold value for edge detection of the object.

The histogram contains two clear peaks representing the object and background intensity values. The image used to generate this histogram contained mostly background values; therefore, the taller peak is located around the mode intensity value of the background, and the lower peak is located around the mode intensity value of the object. The local minimum value between the two peaks is the threshold used for object segmentation. All of the pixels that are below the threshold value are converted to black pixels. Pixel values above the threshold value are converted to white pixels. When this is performed correctly, the object appears white and the background appears black.

When the object and background are separated in the image, some noise filtering may be required in order to attain a clean black and white representation of the image. Once a clean black and white representation of the image is achieved, feature extraction may occur. An example of an image following the image preparation steps is provided in Figure 12.

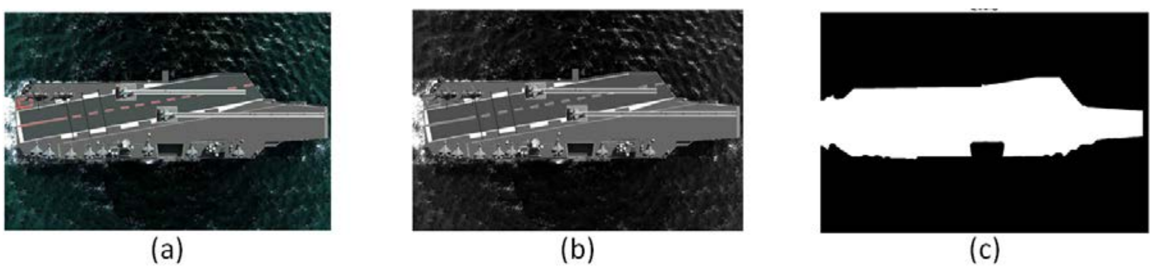


Figure 12. A sample image is shown undergoing the image preparation scheme, where (a) is the original image, (b) is the grayscale image, and (c) is the image of the object segmented from the background.

After an object is segmented from its background, feature extraction is performed. The features include area, perimeter, complexity, spreading, object standard deviation, maximum contrast, mean contrast, maximum gradient, and mean gradient. The feature values are extracted from the grayscale and segmented representations of the image and stored for neural network training preparation. As explained in Chapter II, the eleven feature values from each image are the input values to train the neural network. In Section C of this chapter, we discuss how a neural network trains, tests, and classifies using extracted features.

B. IMAGE PREPARATION FOR HOUGH TRANSFORMATION FEATURES

The Hough transformation is used as an effective and efficient feature extraction tool in this thesis. It is a proven technique that is robust against noise and occlusion [3]. This is an ideal characteristic for target-of-interest detection since maritime domain images usually contain noise from objects of no interest. A process following the object classification scheme using the Hough transformation in Figure 13 is developed in this

section. With the Hough transformation concepts outlined in Chapter II, this process directs how to prepare images and conduct feature extraction.

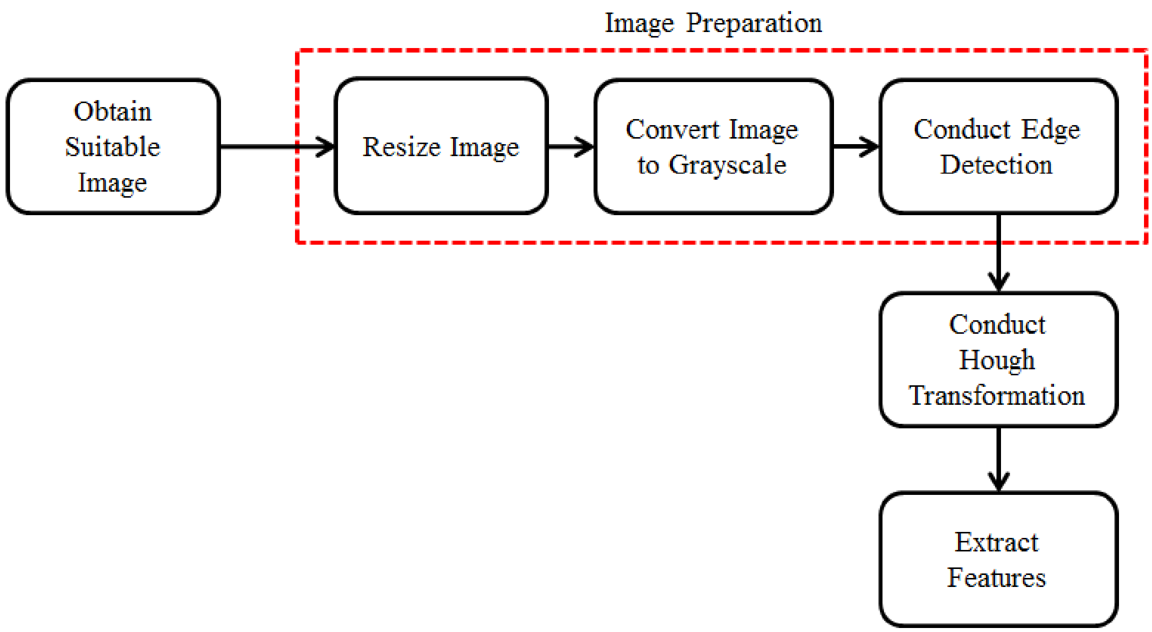


Figure 13. The image preparation scheme encompasses steps to produce an edge representation in order to extract Hough transformation features.

Similar to the extraction of geometric and photometric features, extraction of Hough transformation features cannot be performed on raw image. The first step to the image preparation scheme to exploit the Hough transformation is to resize the image so all of the images are of the same resolution. This step must be performed in order to maintain a fair comparison from image to image. The second step to the image preparation scheme is to convert the image to grayscale. This step is performed to obtain the intensity information of the image. The pixel values are in the range [0, 255] depending on the pixel intensity. This step yields the image exclusively in shades of gray, which varies from black at the weakest intensity to white at the strongest. The third and final step to the image preparation scheme is to conduct edge detection. This step calculates the gradient of the intensity values of the grayscale image. The edges are detected where the gradient of the intensity values are a maximum and returns a white pixel at these maximum gradient locations. A black pixel is returned where there is no

edge detected. An example of an image following the image preparation steps is illustrated in Figure 14.

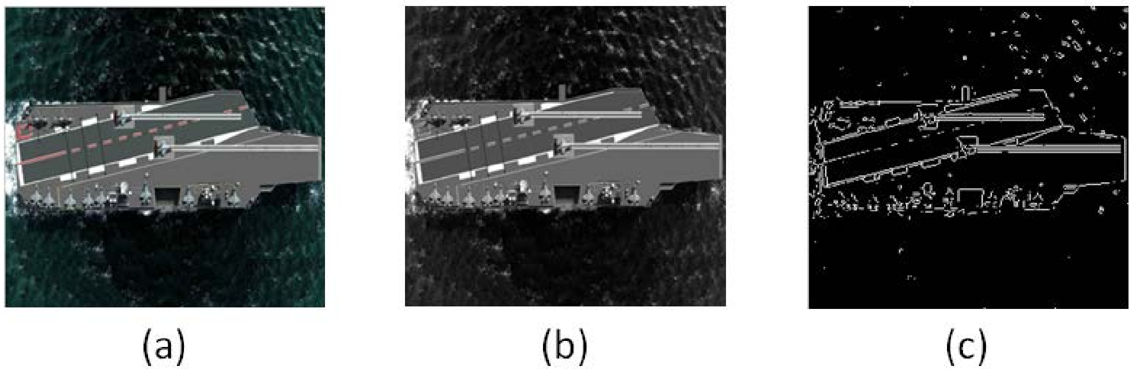


Figure 14. A sample image is shown undergoing the image preparation scheme for the Hough transformation, where (a) is the original image, (b) is the grayscale image, and (c) is the edge detection image.

An image is ready to undergo Hough transformation once it has been processed by the image preparation scheme. After the Hough transformation is conducted on a prepared image, a Hough transformation matrix is generated, called the Hough space. As discussed in Chapter II, the Hough space contains a count of the sinusoidal curve interactions in the (θ, r) polar coordinate plane. The coordinates of the maximum intersection count in the Hough space corresponds to the shape of the object in the image [7]. The first feature extracted is a $N_H \times 1$ vector of the peak values $H_{\max}$ in each column of the Hough space. This feature is a $N_H \times 1$ vector since the Hough space has N_H columns.

The second feature extracted is a vector ρ containing indices corresponding to the peak values $H_{\max}$ in the Hough space. These vectors are determined by $[H_{\max}, \rho] = \max(H)$, where H is the Hough space matrix. This feature is also a $N_H \times 1$ vector since there is one ρ value for each peak value. These two vectors can be combined into one concatenated feature vector of size $2N_H \times 1$. The two sets of features complement each other.

The peak value vector provides characteristics of the dominant straight lines in the image, while the ρ value vector provides characteristics of curvature in the image [3]. The features are proficient in generalizing shapes that are concave or convex in nature [3]. The combination of the two vectors serves an enhanced training tool for the neural network. In the next section we discuss how neural networks are trained using the geometric, photometric, and Hough transformation features and how the neural network is tested for object classification.

C. NEURAL NETWORKS TRAINING, TESTING, AND CLASSIFICATION

The final steps in the object classification scheme are neural network training and testing and classification. The process to select the most effective neural network topology and learning algorithm for binary classification and multi-class classification are explained in this section. The procedure for testing and classifying images is also described in this section.

1. Training a Neural Network for Binary Classification

A series of simulations were performed utilizing various neural network topographies. When the geometric and photometric features were used to train a neural network, the input layer remained at eleven neurons throughout all simulations, which represented the eleven features extracted from the images. The output layer remained at two neurons throughout all simulations, which represented a positive or negative response for a ship present in the image. The hidden layer size was investigated to determine the number of neurons that yield the best classification rate. For the sake of variation, the application of one hidden layer is investigated, and the number of neurons in the hidden layer was varied from 1 to 14 neurons. An example of the topology used for geometric and photometric feature-based simulations is provided in Figure 15.

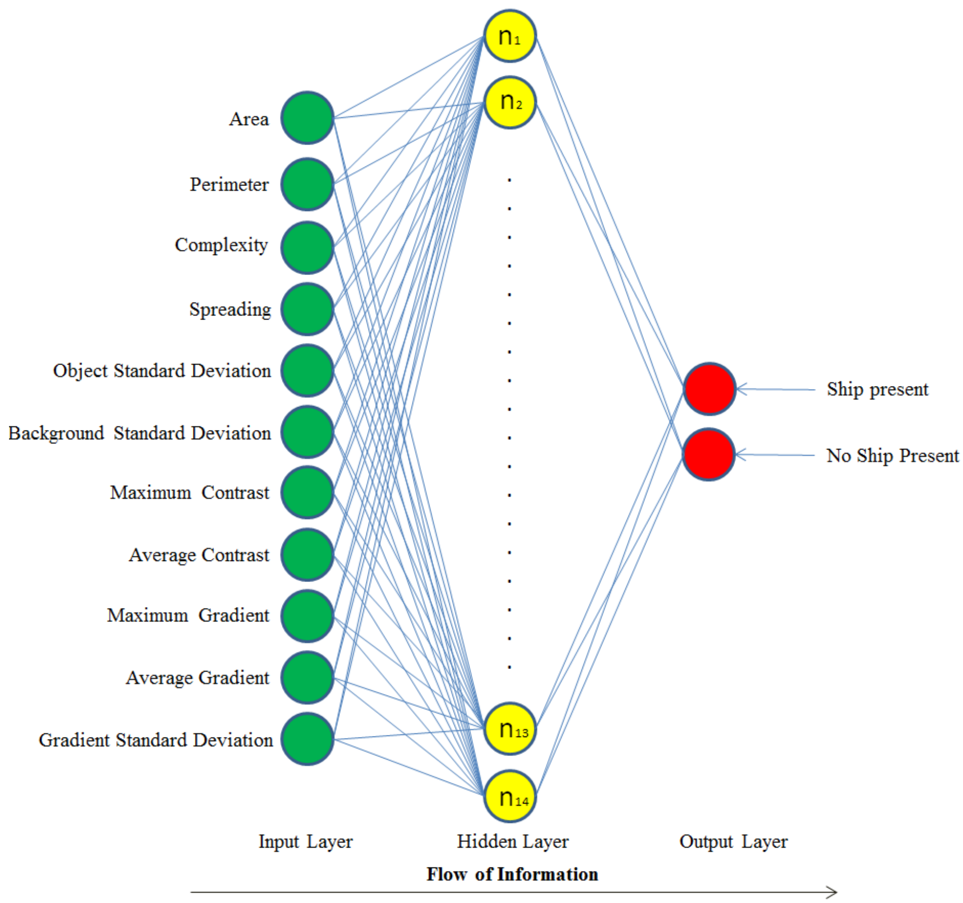


Figure 15. A neural network is designed for binary classification using geometric and photometric features for training. The eleven features are the inputs, with a variable hidden layer size, and two output neurons for a positive or negative response for a ship present in the image.

A neural network is also designed for binary classification using the extracted Hough transformation features. A series of simulations were performed utilizing various neural network topographies. The input layer remained at 360 neurons throughout all simulations, representing the vector of 360 values extracted from the Hough space. The output layer remained at two neurons throughout all simulations, representing a positive or negative response for a ship present in the image. The hidden layer size was investigated to determine the number of neurons that yield the best classification rate. For the sake of consistency, the use of one hidden layer was investigated, and the number of neurons in the hidden layer was varied from 10 to 400 neurons. An example of the topology used for Hough transformation-based simulations is provided in Figure 16.

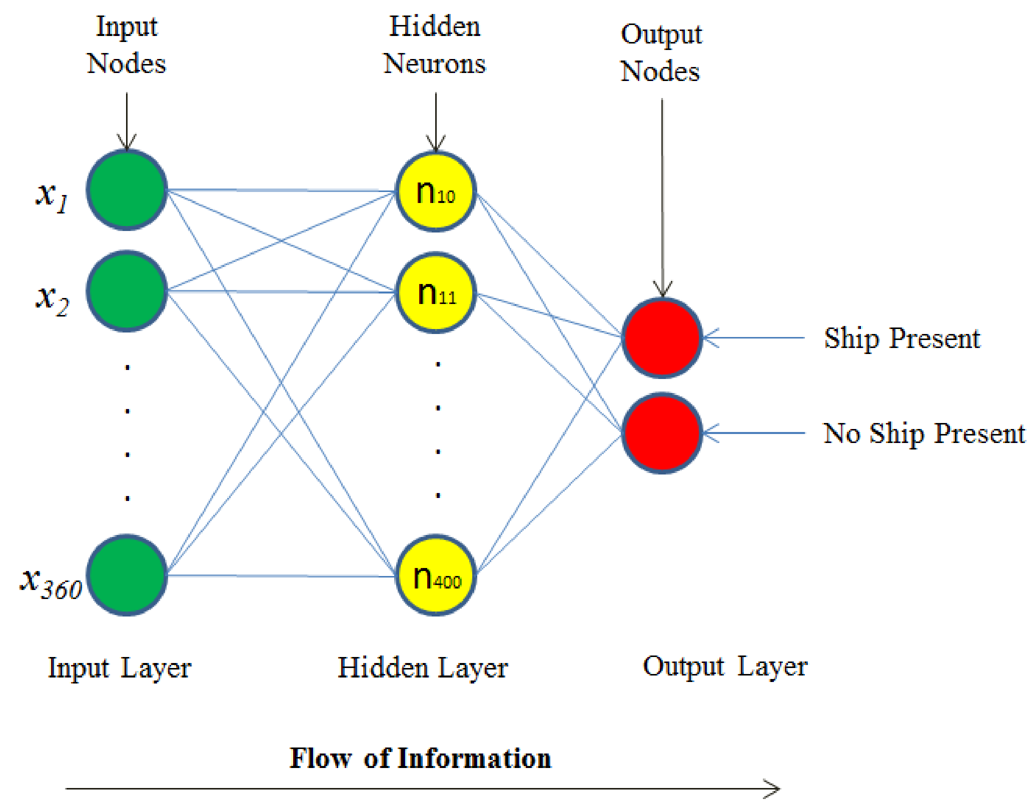


Figure 16. A neural network is designed for binary classification using the features extracted from the Hough space. The vector of 360 values extracted from the Hough space is the input, with a variable hidden layer size, and two output neurons for a positive or negative response to a ship present in the image.

2. Training a Neural Network for Multi-Class Classification

Similar simulations were performed for training a neural network for multi-class classification. These simulations are intended to train a neural network to distinguish between four different classes: warship, cargo ship, small boat, or no ship present. When using the extracted geometric and photometric features, the size of the input layer remained at eleven neurons, representing the eleven extracted features. The hidden layer remained at one layer and the size of the hidden layer was varied from 1 to 14 hidden neurons. The output layer contains four neurons to represent the four different classification possibilities. An example of the topology used for geometric and photometric feature-based simulations for multi-class classification is illustrated in Figure 17.

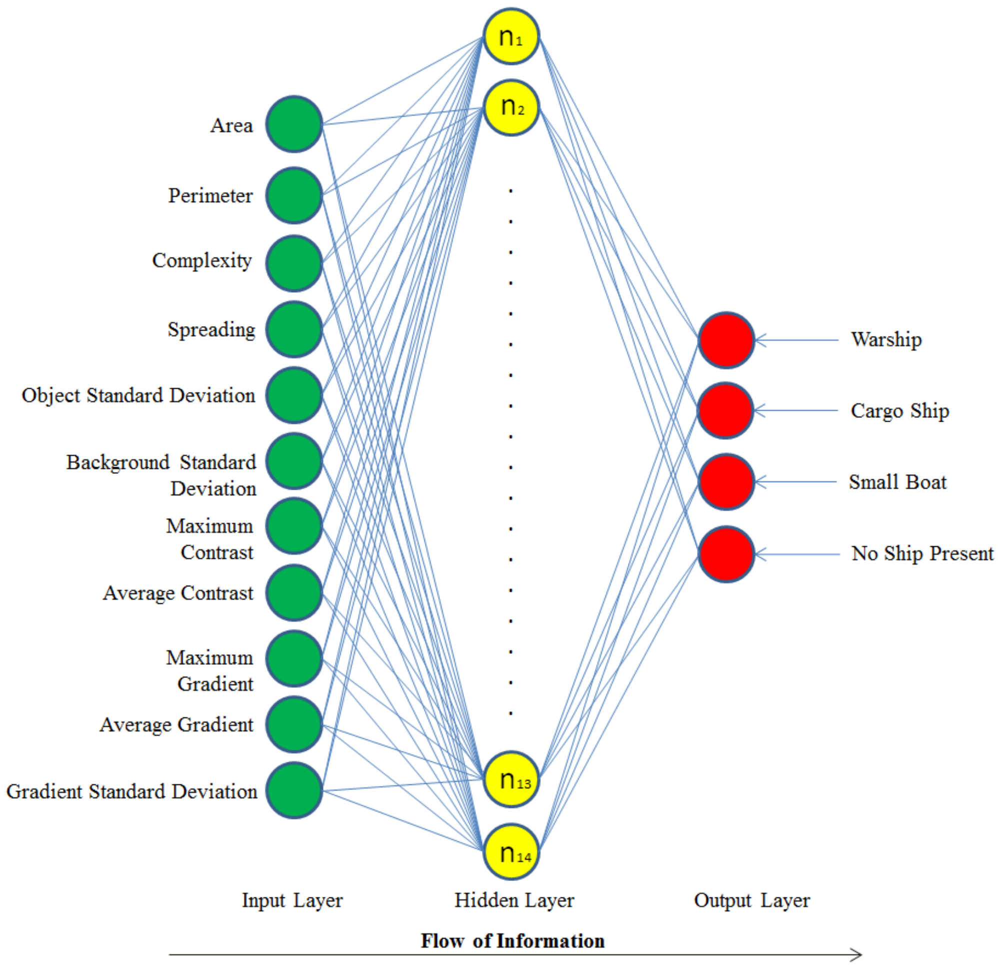


Figure 17. A neural network is designed for multi-class classification using geometric and photometric features for training. The eleven features are the inputs, with a variable hidden layer size, and four output neurons for each possible classification.

Similar simulations were performed using Hough transformation features for training a neural network for multi-class classification. The size of the input layer remained at 360 neurons, representing the vector of 360 values extracted from the Hough space. The hidden layer remained at one layer and the size of the hidden was varied from 10 to 400 hidden neurons; however, the output layer contained four neurons to represent the four different classification possibilities. An example of the topology used for Hough transformation-based simulations for multi-class classification is illustrated in Figure 18.

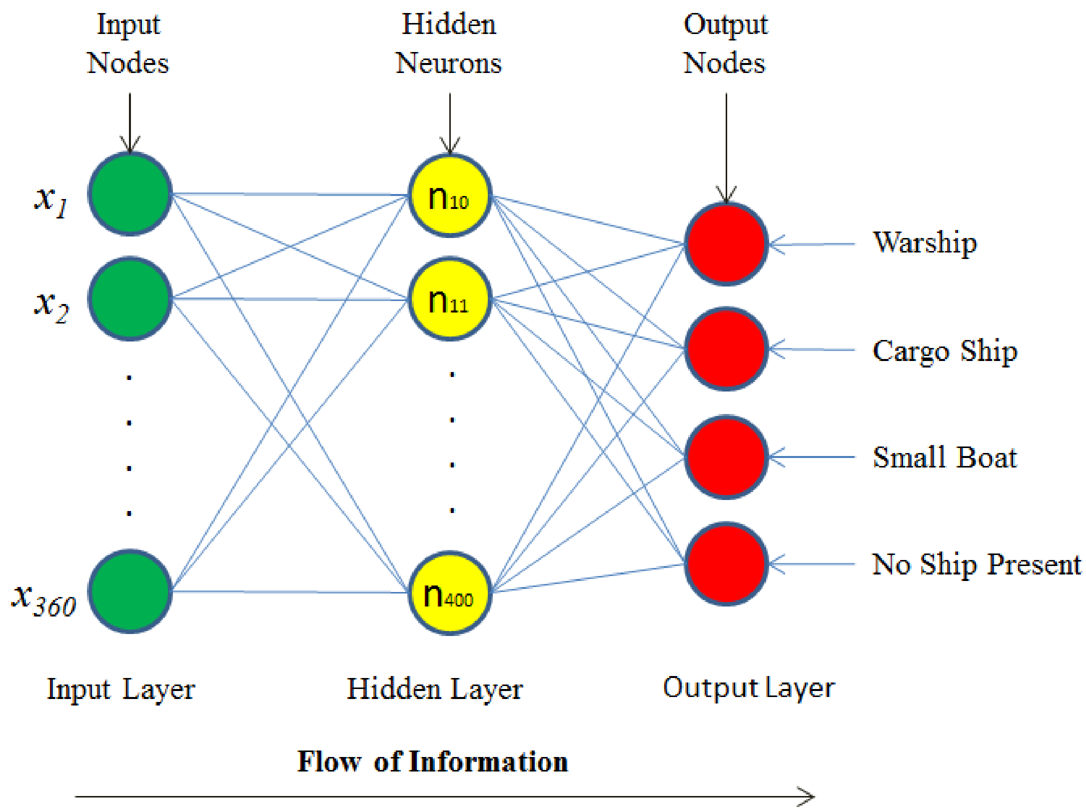


Figure 18. A neural network is designed for multi-class classification using the features extracted from the Hough space. The vector of 360 values from the Hough space is the input, with a variable hidden layer size, and four output neurons for each ship classification.

Learning algorithms are mathematical procedures which drive the weights to provide the correct output in supervised learning [9]. A learning algorithm is used to automatically adjust the neural network's weights during the training phase. Two learning algorithms were used in the simulations to observe which algorithm performed the best in the binary classification and multi-class classification cases: resilient backpropagation and scaled conjugate gradient backpropagation. The algorithms use Equations 8 and 9, respectively, to update the neural network's weights. The number of iterations to achieve optimization was also recorded for each algorithm. More iterations means longer training durations. With large image sets, the time to train a neural network can be time-consuming. Training a neural network to recognize a target-of-interest may require time-sensitive information and demand prompt feedback. The trade space of training time and percent classification must be considered when choosing the best learning algorithm. If

one algorithm has a slight advantage in classifying an image over another but expends twice as long to train, the other algorithm may be selected instead to spare accuracy in favor of training time. In the next section, we discuss how neural networks use training to test and classify an image.

3. Testing and Classification

Once the neural networks have been trained for their respective classification goals, they are ready to be tested with a known sample image. The features of the known image are extracted and are entered into the input layer of the neural networks. The neural networks generalize a classification for the object based on the weights that were formulated during the training phase. A correct classification means that the neural network correctly classified the object in the image. The classification rate for each hidden layer size and learning algorithm pair were collected to observe which combination yielded the best results.

D. FEATURE SELECTION

As explained in Chapter II, a feature selection process is required that differentiates between features of strong or weak relevance. To reiterate, a feature with strong relevance implies that the removal of that feature from the training phase severely degrades classification performance [6]. A feature with weak relevance implies that the removal of that features has little or no impact to the classification performance. Features with strong relevance are recommended to remain as inputs for training a neural network, whereas features of weak relevance may be discarded from the feature set.

The feature selection process assists in determining which features are critical to classifying maritime domain images. This pruning process categorizes the level of significance of each extracted feature. To determine the level of significance of each feature, a neural network was trained with the removal of each feature, and the classification performances were recorded. The features are divided into three different levels of significance based on the relevance. The performance parameters for each significance level are defined in Table 2.

Table 2. The significance for each feature is classified in three different levels according to the defined performance parameters. The most significant features are categorized in Level I. Slightly less significant features are categorized in Level II, and features with little to no significance are categorized in Level III.

Level of Significance	Parameters
I	$\Delta > 4\%$ degradation in classification performance
II	$2\% < \Delta < 4\%$ degradation in classification performance
III	$\Delta < 2\%$ degradation in classification performance or improved performace

In this chapter, we proposed an image classification scheme for maritime domain awareness. The techniques for geometric and photometric features as well as Hough transformation features were discussed. The development of neural networks trained using the extracted features to identify targets-of-interest were described. In the next chapter, we discuss the implementation of the feature extraction techniques in MATLAB, implementation of the neural networks using the MATLAB Neural Network Toolbox, and the target-of-interest classification performance.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS

The methods described in Chapter III are applied to a collection of images, and the results are reported in this chapter. The implementations of these methods, which include programming techniques, are discussed in the first section. The results and recommendations for neural network target-of-interest detection performance are detailed in the second section.

The software tool that was used to perform image processing in this thesis was MATLAB, which supports several image formats, such as .JPEG and .png. MATLAB also contains pre-programmed functions that assist in the conversion of images, plotting pixel intensity values, and segmenting the object from the background. The MATLAB Neural Network Toolbox™ was utilized for the classification of objects used in this thesis. The Neural Network Toolbox supports supervised learning neural networks, which were chosen to best implement object classification. The toolbox provides the capability for designing, training, visualizing, and simulating neural networks, which proved to be a flexible and reliable software package for this research.

A. IMPLEMENTATION

The object classification scheme proposed in Chapter III is implemented in MATLAB and applied to a collection of images to explore the viability of classifying targets-of-interest in the maritime domain using neural networks. The object classification scheme is the basis for experimentation in this thesis and is detailed in Figure 19.

The scheme begins by accumulating a large collection of maritime domain images. In this thesis, a collection of 104 images from the maritime domain was gathered from various sources. Nearly half of the images collected contained ships, while the other half contained other objects commonly found in the maritime domain. The images contained only one object from an overhead aspect angle.

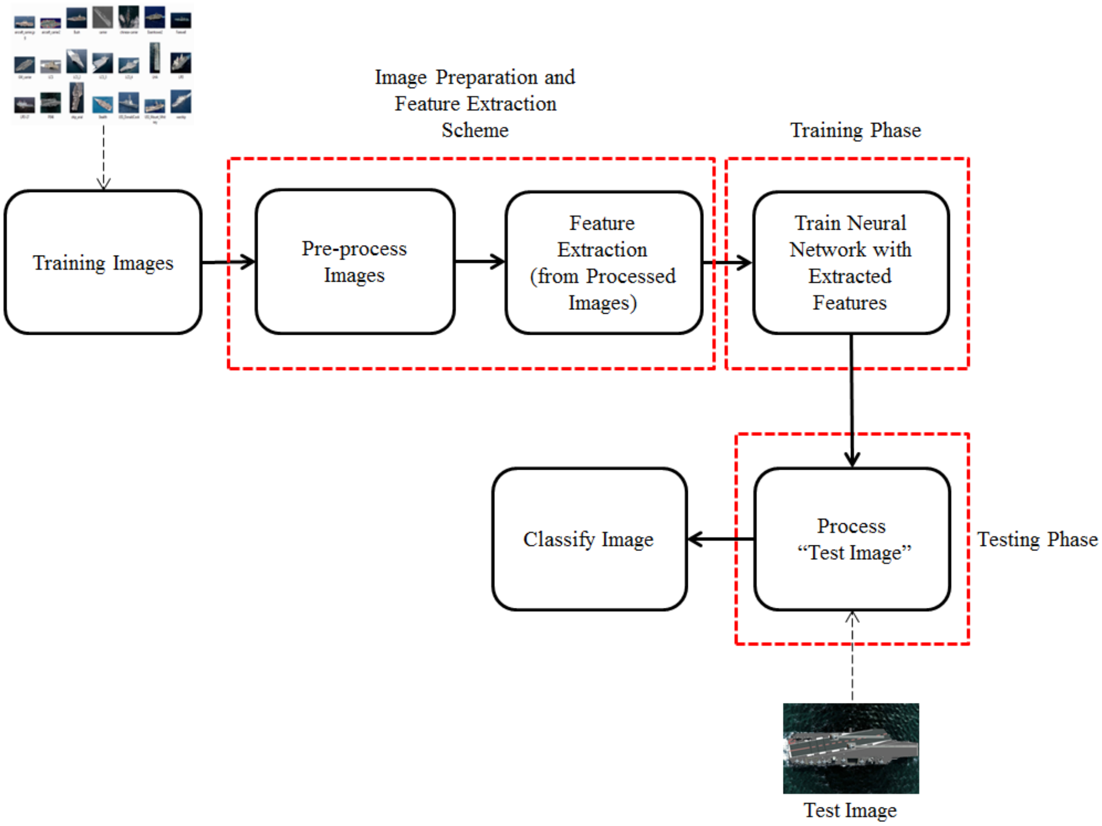


Figure 19. The object classification scheme encompasses the steps to prepare images, extract features, as well as train and test a neural network using images from the maritime domain.

The next step in the object classification scheme is to perform image pre-processing and extract the valuable features from the prepared images. Image preparation and feature extraction results are presented later in this chapter.

Once the features have been extracted, they are presented as inputs to a neural network for training. For the binary and multi-class classification implementations, the input vector to a neural network using geometric and photometric features was a 11×104 matrix: 11 features for 104 maritime domain images. The input vector to a neural network using Hough transformation features was a 360×104 matrix: 360 Hough features for 104 maritime domain images. For the binary and multi-class classification implementations, the neural networks were trained until the mean-squared error reached a small value ε. For the purposes of this thesis, we chose $\varepsilon = 10^{-6}$. The training time and training performance were collected and analyzed. In order to analyze the neural

network's training performance, the number of iterations for the neural network to meet its stopping criterion was assessed, i.e., the training iterations were continued until the mean-squared error reached ε. When the training concluded, the weights of the neural network were fixed, and the neural network was ready to process an image for testing.

The conventional approach to testing is to obtain results from a single neural network. This neural network is trained with a conservative sample size of the image set. A separate set of images is then processed and used for testing. Since the collection of images in this thesis was too small to properly train a neural network, testing was performed differently from the conventional approach. To achieve more reliable results, 103 images were used to train a neural network, and the remaining image was used for testing. This procedure was repeated for 1000 trials using a randomly chosen 103 images for each trial. One thousand trials were performed with various neural network topologies and learning algorithm combinations to determine the combination that yielded the best classification performance.

Binary classification implementation in this thesis separates ship images from other images. Examples of images that are classified under "other" are images of clouds, whales, aircraft, etc. Binary classification performance is determined by the number of correct ship image classifications over the 1000 trials. The breakdown of the image set for binary classification is provided in Table 3.

Table 3. Breakdown of the images for binary classification.

Type	Number of Images
Ship Images	54
Other	50

Multi-class classification implementation in this thesis separates the image collection into four classes: warship, cargo ship, small boat, and other. Images that are classified under "warship" are images of any vessel of war, such as an aircraft carrier. Examples of images that are classified under "cargo ship" include images of oil tankers, container ships, cruise liners, etc. "Small boat" images include images of yachts, fishing

vessels, dhows, etc. "Other" images include objects commonly found in the maritime domain other than ships as previously stated. The targets-of-interest for the purposes of this thesis are images containing warships; multi-class classification performance is determined by the number of correct warship image classifications of the 1000 trials. The breakdown of the image set for multi-class classification is provided in Table 4.

Table 4. Breakdown of the images for multi-class classification.

Type	Number of Images
Warship	22
Cargo Ship	22
Small Boat	10
Other	50

A multi-level approach to multi-class classification was developed as well. The multi-class classification trains a neural network to separate the object classes in one decision-making step. In the multi-level approach, a series of binary classifications are performed to separate the object classes. Similar to multi-class classification, multi-level classification performance is determined by the number of correct warship image classifications. In Figure 20, the decision-making process for each approach is illustrated.

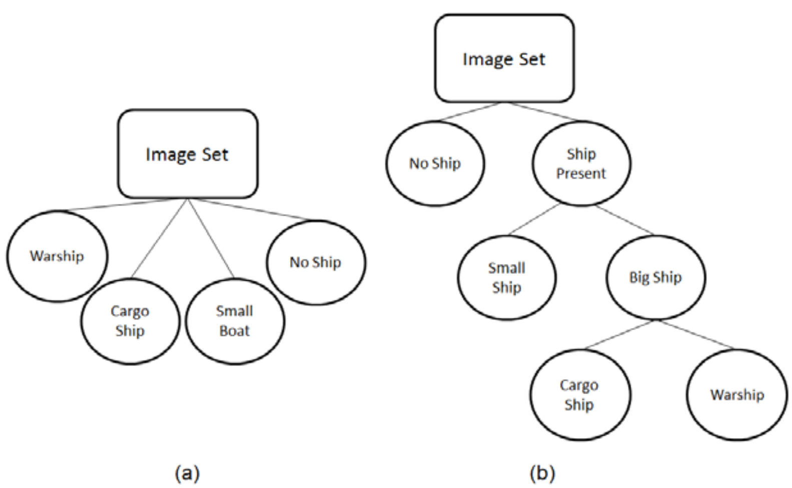


Figure 20. Two approaches to separating object classes are developed. (a) Multi-class classification separates object classes in one decision-making step. (b) Multi-level classification uses a series of binary classifications to separate object classes.

The results for each of the classification schemes are presented in the following sections. The results for each feature extraction technique are presented first, followed by the results of neural network training and testing.

B. FEATURE EXTRACTION

Two feature extraction techniques are developed in this thesis. The first technique segments the object from the background and extracts geometric and photometric features. The second technique utilizes the Hough transformation to extract features of the object's shape. The results of each feature extraction technique are presented in this section.

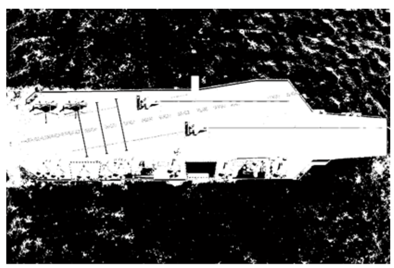
1. Image Preparation for Geometric and Photometric Feature Extraction

As described in Chapter III, the raw maritime domain images require processing in order to retrieve valuable information. The MATLAB code used to perform image processing to segment the objects from their background is included in Appendix A. The first step for image preparation is to convert each of the images to grayscale. MATLAB's built in *rgb2gray* function [13] is used to perform this operation. Essentially, *rgb2gray* converts the true color of the image to shades of gray depending on the intensities of the image. The intensity values range from 0 to 255 to represent every pixel value for an eight bits per pixel image. Once the grayscale images are obtained, the pixel intensities are normalized to yield values ranging from 0 to 1.

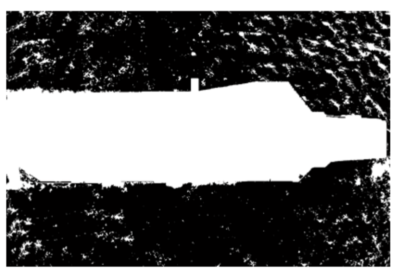
The next step in the image preparation scheme is to segment the object in the image from the background. This is accomplished by first plotting a histogram of the normalized pixel intensity values. MATLAB's built in *imhist* function [14] performs this operation. The histogram contains two clear peaks representing the object and background. The images contain mostly intensity values of the background, which creates a peak located around the intensity value of the background. The smaller peak is located around the intensity value of the object. The local minimum value between the two peaks is the threshold used for object segmentation. The threshold value is computed as the midpoint of the two peaks in the histogram. MATLAB's built in *findpeaks* function [15] is used to find the location of the peaks. All of the pixels that are below the threshold

are made black, and the pixel values above the threshold are made white. This can be performed using the MATLAB's *im2bw* function [16], which uses the threshold value and converts the image to black and white. When this is performed correctly, the object appears white and the background appears black.

The final step in image preparation is to perform noise filtering. In some cases, intensity values of the object are shared with regions of the background. This is detected when regions of the background appear white. Examples of segmented images that require additional noise filtering is illustrated in Figure 21. MATLAB's built in functions, such as *imerode* and *imfill*, are used to remove white regions in the object and fill in black regions in the object. These functions are fully detailed in [17] and [18].



(a)



(b)

Figure 21. A pair of segmented images that require additional noise filtering, where (a) requires the *imfill* function to fill in black regions of the object, and (b) requires the *imerode* function to remove white regions from the background.

2. Geometric and Photometric Feature Extraction

Now that the objects in all images have been segmented from the background, the geometric and photometric features are extracted. The following is a review of the MATLAB functions used to extract the feature values. The MATLAB code used to perform feature extraction is included in Appendix B.

The area (A) of segmented objects in every image is computed using MATLAB's built in *bwarea* function [19]. This function counts the number of white pixels within the image.

The perimeter (P) of segmented objects in every image is outlined using MATLAB's built in *bwperim* function [20]. This function returns a black and white image only containing the outline of the object in white. The perimeter value is calculated by counting the number of white pixels in the perimeter image.

The complexity (C) feature is defined as a function of the area and perimeter of the object in the image. The determination of C is calculated using Equation 1.

The spreading (S) feature is defined as level of "spreading" the object has in the image. The determination of S is calculated using Equation 2. The eigenvalues λ_1 and λ_2 are associated with the covariance matrix of the processed image [4]. The covariance matrix is generated using the MATLAB's *cov* function [21], and the eigenvalues of the covariance matrix were calculated using MATLAB's *eig* function [22].

The object standard deviation (σ_{obj}) feature is defined as the standard deviation (in dB) of pixel intensity values belonging to the object. This is obtained by computing the standard deviation of the object's intensity values using MATLAB's built in *std* function [23].

The background standard deviation (σ_{back}) feature is defined as the standard deviation (in dB) of pixel intensity values belonging to the background. This is determined by computing the standard deviation of background intensity values using the *std* function and then converting the standard deviation to dB [4].

The maximum contrast (κ_{max}) feature is defined as the difference (in dB) between the average background intensity value and the lowest intensity value of the object [4]. The average background intensity value was calculated using MATLAB's *mean* function [24], and the lowest intensity value of the object was found using MATLAB's *min* function [25].

The mean contrast (κ_{av}) feature is defined as the difference (in dB) between the average background intensity value and the average object intensity value [4]. The averages of the background and object intensity values were calculated using the *mean* function, and the difference between the values were computed.

The maximum gradient (δ_{max}) feature is defined as the maximum value (in dB) of the border gradient. This was determined by computing the gradient of the intensity values for the object's border using MATLAB's *gradient* function [26]. The maximum gradient of the border was attained using MATLAB's *max* function [27] and then represented in dB [4].

The mean gradient (δ_{av}) feature is defined as the average border gradient represented in dB. This was determined by computing the gradient of the intensity values for the object's border using the *gradient* function. The average of the border gradient was attained using the *mean* function and then represented in dB [4].

The gradient standard deviation (σ_{grad}) feature is defined as the standard deviation of the border gradient values in dB [4]. This was determined by computing the gradient of the intensity values for the object's border using the *gradient* function. The standard deviation of the border gradients was attained using the *std* function and then represented in dB [4].

The statistical parameters of the features extracted from ship and non-ship images are provided in Table 5. Although the statistical parameters provided in Table 5 were not used in the training of the neural network, they are useful to examine which features can clearly define ship images from images of no ship present. Later, the significance of each feature is investigated using a pruning process.

Table 5. The statistical parameters describing the extracted features from ship and non-ship images. When determining relevant features, it is useful to examine features that distinguish ship images from non-ship images.

	Ship				Non-Ship			
Feature	**Min**	**Max**	**Mean**	**Std Dev**	**Min**	**Max**	**Mean**	**Std Dev**
Area	2437	2726684	149270	408448	1831	3666184	340941	714606
Perimeter	315	82469	3397	11150	188	60764	4701	9333
Complexity	1.23	14.09	2.31	1.77	1.04	8.95	2.54	1.53
Spreading	0.93	17.88	5.04	3.41	1.19	47.74	26.19	11.41
Object Standard Deviation	5.35	13.18	7.30	1.27	5.79	15.85	9.92	2.18
Background Standard Deviation	6.67	19.15	10.02	2.52	6.84	25.04	11.37	3.17
Maximum Contrast	2.87	27.55	7.88	3.89	0.12	23.38	10.69	5.15
Average Contrast	1.69	9.23	4.65	1.78	1.98	9.73	5.17	1.51
Maximum Gradient	6.23	21.00	10.92	3.01	6.56	38.59	21.61	7.69
Average Gradient	16.93	33.66	24.58	3.71	22.70	48.59	35.21	6.74
Gradient Standard Deviation	14.91	35.73	21.31	3.80	19.03	47.92	32.43	7.70

To implement the binary classification problem in the Neural Network Toolbox, the input vectors, or features, must be arranged as columns in a matrix. A 2×2 matrix of output values specify the image's classification. An example of two input vectors is provided below in Table 6.

Table 6. Example input feature vectors for classifying a ship present or not present.

Feature	Aircraft Carrier	Cloud
Area	15330	2559252
Perimeter	1168	18864
Complexity	2.66	3.33
Spreading	0.98	38.84
Object Standard Deviation	7.98	10.75
Background Standard Deviation	7.72	10.69
Maximum Contrast	4.31	11.30
Average Contrast	5.61	4.03
Maximum Gradient	11.34	31.98
Average Gradient	25.87	46.87
Gradient Standard Deviation	21.89	45.74

From Table 6, the input and output matrices for the example data are as follows:

$$X = \begin{bmatrix} 15330 & 2559252 \\ 1168 & 18864 \\ 2.66 & 3.32 \\ 0.9787 & 38.8411 \\ 7.9807 & 10.7452 \\ 7.7235 & 10.6938 \\ 4.3087 & 11.2964 \\ 5.6106 & 4.0317 \\ 11.33778 & 31.9766 \\ 25.8667 & 46.8731 \\ 21.8891 & 45.7443 \end{bmatrix}$$

$$Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

where the first column in the input matrix contains the features corresponding to the "Aircraft Carrier" image and the second column the features corresponding to the "Cloud" image. The columns in the output matrix indicate to which class the image belongs. The first column indicates that the "Aircraft Carrier" features belong to the "Ship Image" class, and the second column indicates that the "Cloud" features belong in the "Other" class.

Similarly, to implement the multi-class classification problem in the Neural Network Toolbox, the input vectors, or features, must be arranged as columns in a matrix. An example of four input vectors is provided in Table 7.

Table 7. Example input feature vectors for a warship, cargo ship, small boat, and non-ship classification is shown.

Feature	Aircraft Carrier	Tanker	Yacht	Cloud
Area	15330	3634	45818	2559252
Perimeter	1168	445	1182	18864
Complexity	2.66	2.08	1.56	3.33
Spreading	0.98	10.79	4.38	38.84
Object Standard Deviation	7.98	13.18	7.34	10.75
Background Standard Deviation	7.72	19.15	14.05	10.69
Maximum Contrast	4.31	27.55	4.68	11.30
Average Contrast	5.61	7.75	5.82	4.03
Maximum Gradient	11.34	18.95	12.92	31.98
Average Gradient	25.87	33.66	25.77	46.87
Gradient Standard Deviation	21.89	29.54	23.87	45.74

From Table 7, the input and output matrices for the example data are as follows:

$$X = \begin{bmatrix} 15330 & 3634.1 & 45818 & 2559252 \\ 1168 & 445 & 1182 & 18864 \\ 2.66 & 2.0824 & 1.5577 & 3.33 \\ 0.9787 & 10.7868 & 4.3782 & 38.84 \\ 7.9807 & 13.1804 & 7.3379 & 10.75 \\ 7.7235 & 19.1485 & 14.0496 & 10.69 \\ 4.3087 & 27.5504 & 4.6843 & 11.30 \\ 5.6106 & 7.7507 & 5.8222 & 4.03 \\ 11.33778 & 18.9505 & 12.9233 & 31.98 \\ 25.8667 & 33.6563 & 25.7734 & 46.87 \\ 21.8891 & 295442 & 23.8703 & 45.74 \end{bmatrix}$$

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where the first column in the input matrix contains the features corresponding to the "Aircraft Carrier" image. The second column contains the features corresponding to the "Tanker" image. The third column contains the features corresponding to the "Yacht"

image. The fourth column contains the features corresponding to the “Cloud” image. The columns in the output matrix indicate the class to which the image belongs. The first column indicates that the “Aircraft Carrier” features belong in the “Warship” class. The second column indicates that the “Tanker” features belong in the “Cargo Ship” class. The third column indicates that the “Yacht” features belong to the “Small Boat” class. The fourth column indicates that the “Cloud” features belong to the “Other” class.

3. Feature Selection

Feature selection is determined by a pruning process for each of the features. A neural network is trained and tested with the removal of one of the features. The correct classification percentage is recorded and subtracted from the classification percentage when all features are used for training. This process is repeated for each feature to determine the features of strong or weak relevance, and each of the features is categorized into one of the three significance categories, as described in Chapter III. The results of the pruning process are presented in Figure 22.

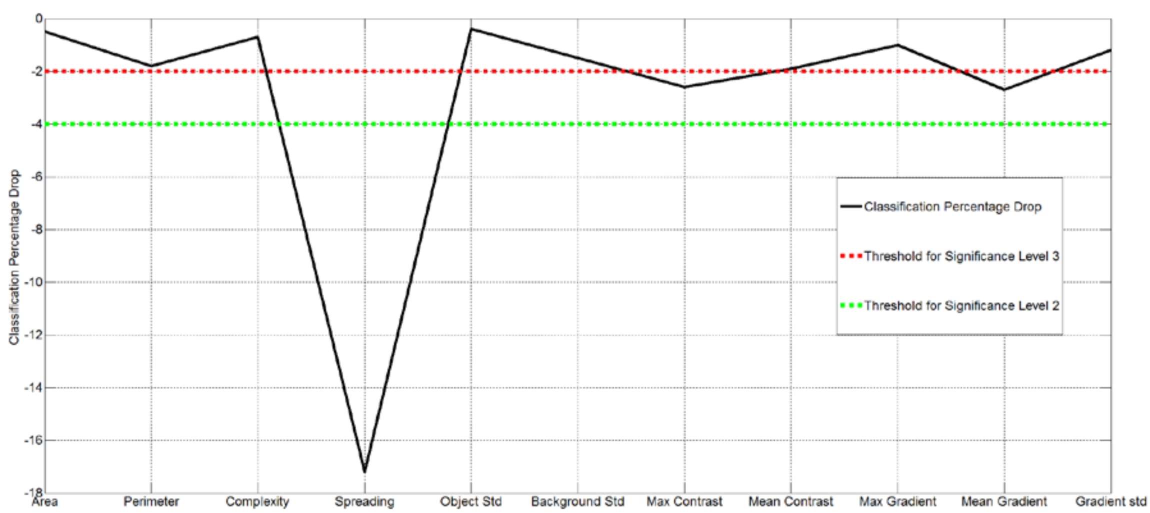


Figure 22. The pruning process establishes which features are useful for training neural networks and classifying objects. A large drop in classification performance signifies a feature is vital for training. Classification percentage drops below the green line denotes Significance Level I. Classification percentage drops between the red and green lines denotes Significance Level II. Classification percentage drops above the red line denotes Significance Level III.

From Figure 22, it can be seen that the Spreading (S) feature had the strongest relevance. When S was removed from training, the neural network declined in classification by nearly 18 percent. It is concluded that the Spreading (S) feature contains the most useful information when classifying a ship present or not present in an image. The remaining features are categorized in Table 8 based on the level of significance parameters defined in Chapter III. The feature pruning process, however, is not used in the simulations presented in the following sections. It is presented here as a potential technique for speeding up the training process and is left for a future effort to verify this advantage.

Table 8. Features are categorized in three different Levels of Significance. The most significant feature is listed in Level of Significance I, less significant features are listed in Level of Significance II, and least significant features are listed in Level of Significance III.

Level of Significance	Features
I	S
II	$\kappa_{max}, \delta_{av}$
III	$A, P, C, \sigma_{obj}, \sigma_{back}, \kappa_{av}, \delta_{max}, \sigma_{grad}$

4. Image Preparation for Hough Transformation Feature Extraction

As described in Chapter III, the Hough transformation can be used to collect valuable information from images for training a neural network. The same 104 maritime domain images are used for the implementation of the Hough transformation techniques. The Hough transformation techniques detailed in Chapter III were used to extract features from the Hough space. The MATLAB code used to perform feature extraction from the Hough space is included in Appendix C.

The first step to image preparation is to resize all of the images so that they all have the same resolution. It is important to analyze images of the same resolution so that

the features are compared equally. MATLAB's built in *imresize* function [28] is used to resize the images to a size of 256×256. The next step to prepare images is to convert each of the images to grayscale as explained previously.

The final step to prepare images is to conduct edge detection. MATLAB's built in *edge* function [29] is used to perform this operation. The *edge* function uses the intensity values from the grayscale image to calculate the gradient from pixel to pixel. The edges are located where pixel gradients are a maximum, corresponding to a white pixel. All other pixels are displayed as black, providing a binary, black and white image.

5. Hough Transformation Feature Extraction

The image is ready to be processed by the Hough transformation once the edges of the object are located. MATLAB's built in *hough* function [30] is used to perform this operation. Essentially, *hough* returns the Hough space. The Hough transformation features are extracted using the values from the Hough space. The first feature extracted is the vector of peak values. From $[H_{\max}, \rho] = \max(H)$, two column vectors, $H_{\max}$ and ρ, are returned. The $H_{\max}$ vector contains the maximum values from each of the columns in the Hough space. Since there are $N_H = 180$ columns in the Hough space, the resulting vector is a 180×1 vector. The $H_{\max}$ values are normalized to be in the range of zero to one.

The ρ vector contains the indices corresponding to the $H_{\max}$ values and is the second feature extracted from the Hough space. Since there are $N_H = 180$ columns in the Hough space, the resulting vector is a 180×1 vector. The ρ values were also normalized to be in the range of zero to one.

The $H_{\max}$ and ρ vectors were combined, creating a 360×1 vector for each image. The vectors from each image were then concatenated to form a 360×104 matrix that serves as the training input for a neural network.

C. NEURAL NETWORK TRAINING

Once the input and output vectors for each image are defined, they are ready to be applied to a neural network for training. The training time and training performance were collected and analyzed. In order to analyze the neural network's performance during the training phase, we assessed the number of iterations for the neural network to converge to the stopping criterion for each of the learning algorithms. The following is a presentation of the performance results of neural networks during the training phase for binary and multi-class classification implementations. The MATLAB code used to perform neural network training with MATLAB's Neural Network Toolbox is included in Appendix D.

1. Binary Classification

The goal of binary classification is to separate the ship images from the non-ship images. The performance for each of the learning algorithms using geometric and photometric features for binary classification is presented in Figure 23. We also assessed how the size of the hidden layer affects the neural network as it converges to the stopping criterion ε. The convergence performance for each learning algorithm as the hidden layer size is varied from 1 to 14 neurons is presented in Figure 24.

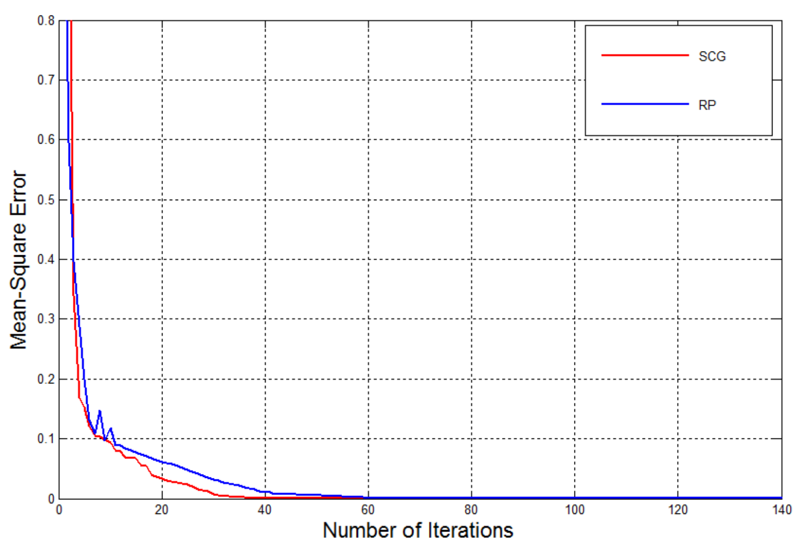


Figure 23. The number of iterations to converge to ε using geometric and photometric features for binary classification. Fewer iterations lead to faster training phase.

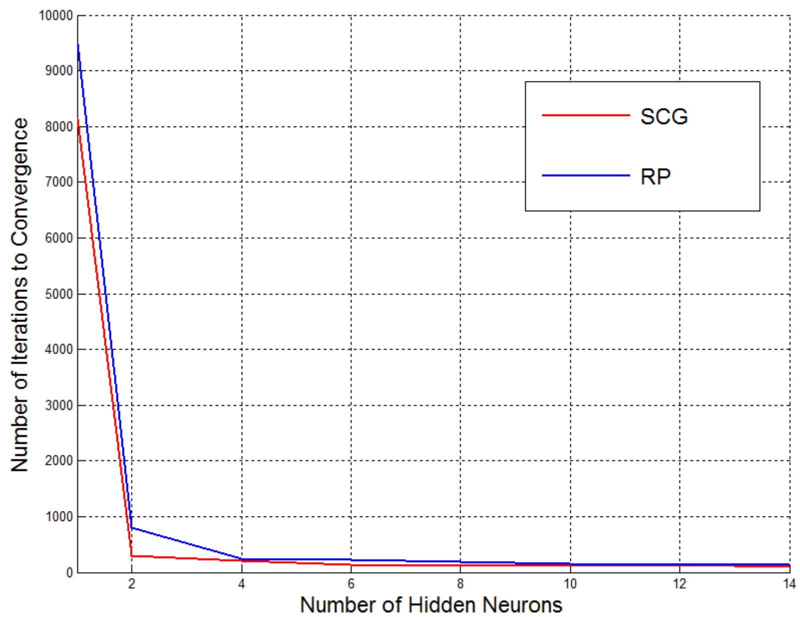


Figure 24. The number of iterations to converge to ε for each topology and learning algorithm combination during the training phase. Geometric and photometric features are used for training a binary classification neural network. As the size of the hidden layer increases, the number of iterations to convergence decreases.

From Figure 23, it can be seen that the scaled conjugate gradient backpropagation algorithm (SCG) reached ε in the smallest number of iterations. For the binary classification application, this learning algorithm trains the neural network the fastest. It is important to remember that the time to train a neural network is important when dealing with time-sensitive information.

From Figure 24, it can be seen that the SCG algorithm converges to ε with the smallest number of iterations for every hidden layer size. Another key observation from this plot is the number of iterations required for convergence approaches a constant convergence rate when the hidden layer reaches eleven or more neurons. This observation has an impact on the recommended topology for binary classification discussed in Section D.

Similar simulations are performed using the extracted Hough transformation features. The performance for each of the learning algorithms using Hough transformation features for binary classification is presented in Figure 25. We also

assessed how the size of the hidden layer affects the neural network as it converges to the stopping criterion ε. In Figure 26, each learning algorithm's convergence performance is presented as its hidden layer size is varied from 10 to 400 neurons.

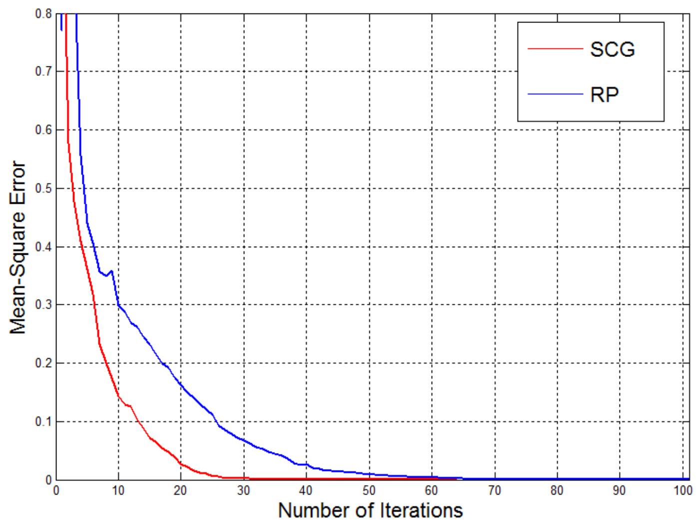


Figure 25. The number of iterations to converge to ε using Hough transformation features for binary classification. Fewer iterations lead to faster training phase.

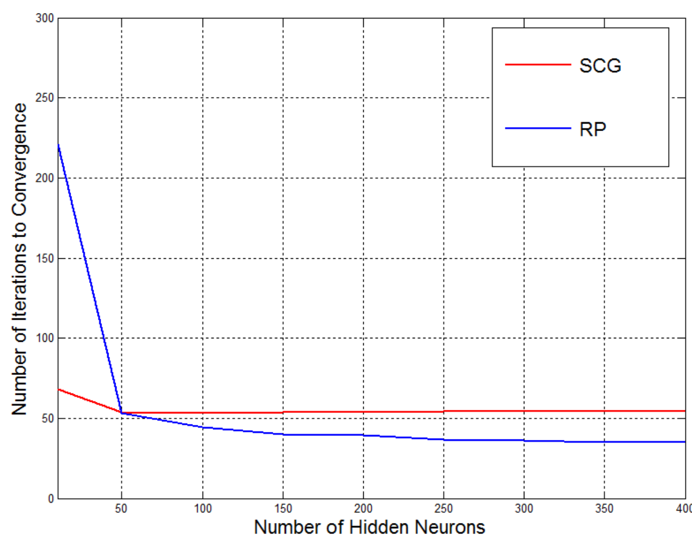


Figure 26. The number of iterations to converge to ε for each topology and learning algorithm combination during the training phase. Hough transformation features are used for training a binary classification neural network. As the size of the hidden layer increases, the number of iterations to convergence decreases.

From Figure 25, it can be seen that the SCG algorithm reached ε in the smallest number of iterations. For the binary classification application, this learning algorithm trains the neural network the quickest.

From Figure 26, it can be seen the resilient backpropagation algorithm (RP) converges the fastest for most of the neural network topology combinations. It is important to remember that the time to train a network is important when dealing with time-sensitive information; therefore, the time to train a neural network is considered when recommending a learning algorithm and hidden layer size combination for binary classification. Another observation from the Figures 23-26 is the comparison in number of iterations required to reach ε using geometric and photometric features versus Hough transformation features. The number of iterations to converge using Hough transformation features is significantly smaller than the number of iterations for neural networks trained with geometric and photometric features. This is a valuable characteristic of using Hough transformation features during training.

2. Multi-Class Classification

The goal of multi-class classification is to separate the collection of images into four different object classes. Similar to the binary classification case, the neural network's performance during the training phase was analyzed by assessing the number of iterations required for convergence. The performance for each of the learning algorithms using geometric and photometric features for multi-class classification is presented in Figure 27. We also assessed how the size of the hidden layer affects the neural network as it converges to the stopping criterion ε. In Figure 28, each learning algorithm's convergence performance is presented as its hidden layer size is varied from 1 to 14 neurons.

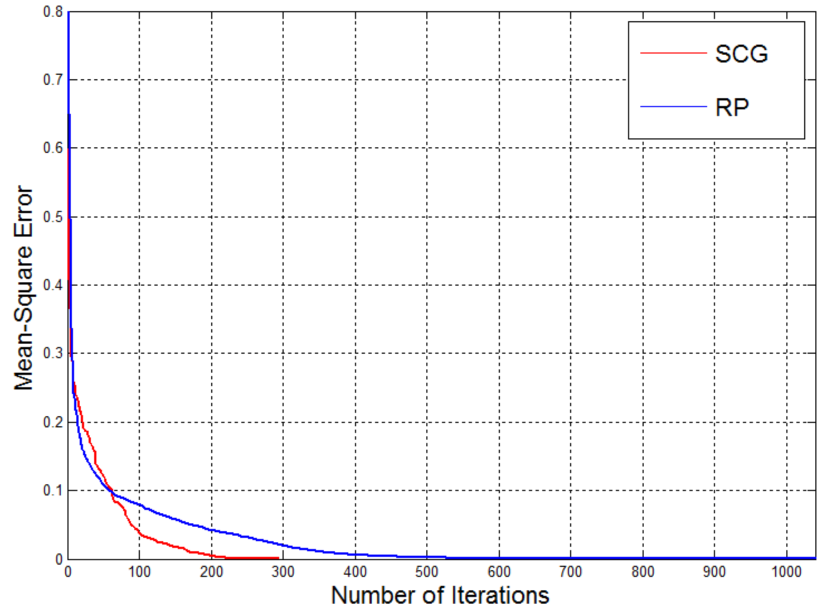


Figure 27. The number of iterations to converge to ε using geometric and photometric features for multi-class classification. Fewer iterations lead to faster training phase.

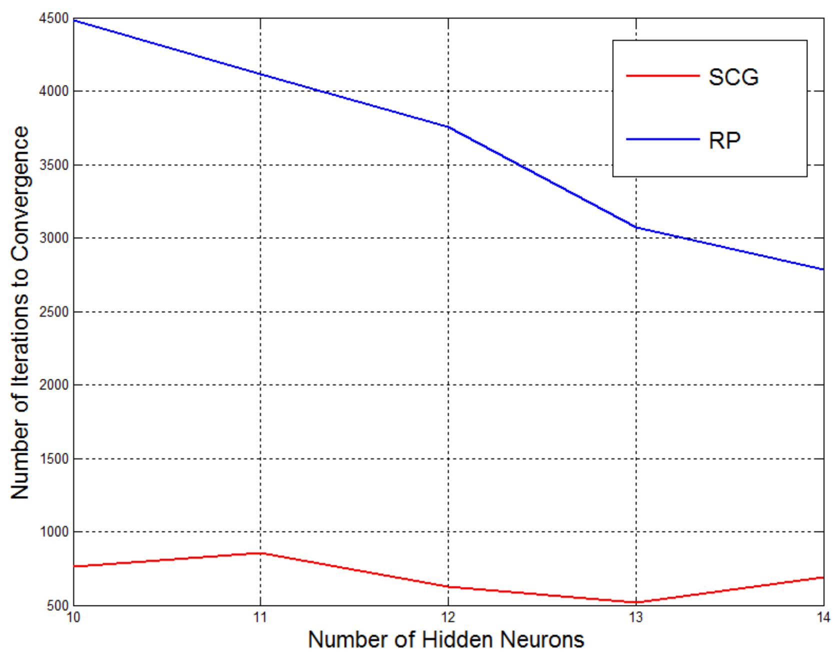


Figure 28. The number of iterations to converge to ε for each topology and learning algorithm combination during the training phase is shown. Geometric and photometric features are used for training a multi-class classification neural network. As the size of the hidden layer increases, the number of iterations to convergence decreases.

From Figure 27, it can be seen that the SCG algorithm again reached the desired mean-square error for the smallest number of iterations. For the multi-class classification application, this learning algorithm trains the neural network the fastest.

From Figure 28, it can be seen that the SCG algorithm converges the fastest for every hidden layer size. It can also be seen that the number of iterations required for convergence is steady when the hidden layer size reaches eleven or more hidden neurons.

Similar simulations are performed using the extracted Hough transformation features. The performance for each of the learning algorithms for multi-class classification using the Hough transformation as a training feature is presented in Figure 29. We also assessed how the size of the hidden layer affects the neural network as it converges to the stopping criterion ε. The convergence performance for each learning algorithm as the hidden layer size is varied from 10 to 400 neurons is presented in Figure 30.

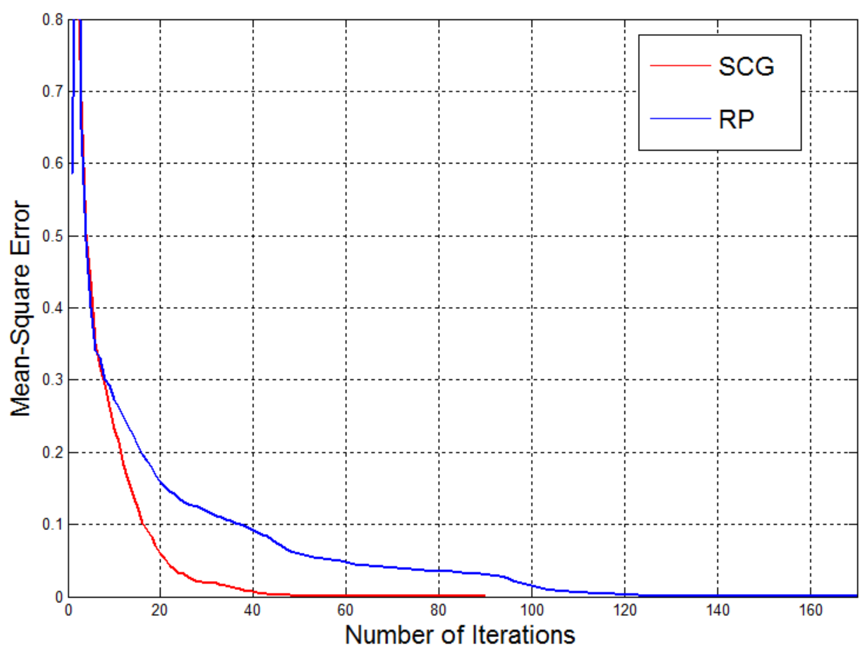


Figure 29. The number of iterations to converge to ε using Hough transformation features for multi-class classification. Fewer iterations leads to faster training phases.

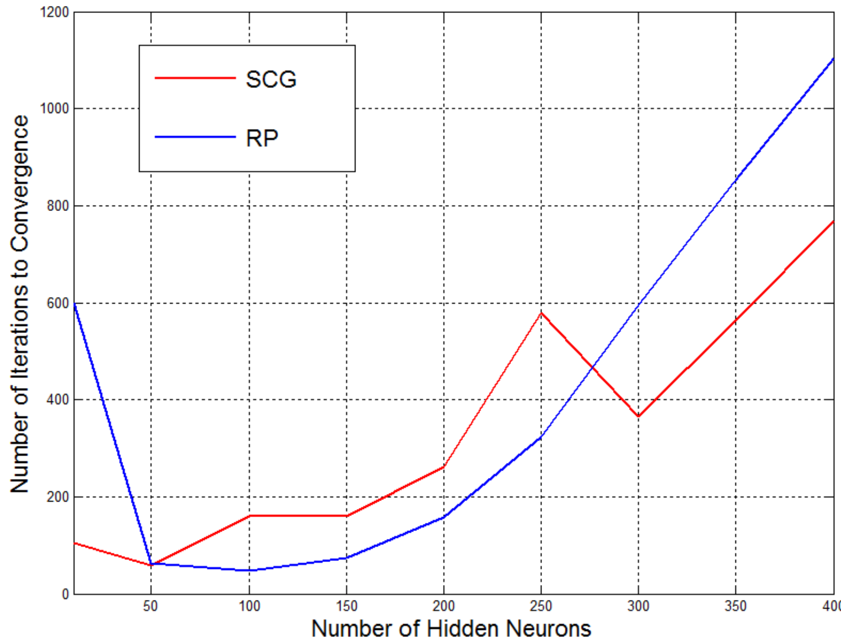


Figure 30. The number of iterations to converge to ε for each topology and learning algorithm combination during the training phase. Hough transformation features are used for training a multi-class classification neural network. As the size of the hidden layer increases, the number of iterations to convergence increases.

From Figure 29, it can be seen that the SCG algorithm reached the desired mean-square error in the fewest iterations. For the binary classification application, this learning algorithm trains the neural network the fastest.

From Figure 30, it can be seen the convergence worsens as the hidden layer size is increased. The time to train a neural network, along with the classification performance, plays an important role when recommending a learning algorithm and topology combination for multi-class classification. Another observation from the Figures 27–30 is the comparison of the number of iterations required to converge using geometric and photometric features versus Hough transformation features. The number of iterations for convergence using Hough transformation features is significantly smaller than the number of iterations for neural networks trained with geometric and photometric features. This is an advantage of using Hough transformation features during training.

D. NEURAL NETWORK TESTING AND CLASSIFICATION

Now that the neural networks have been trained for their respective classification objectives, they are tested. The neural networks use the weights developed in the training phase to classify the remaining image in each of the simulations. The classification performance was calculated as a percentage of the number of correct object classifications during 1000 trials. Of the 104 images considered for testing, 54 were ship images and 50 were non ship images. The following is a presentation of the testing and classification results for binary classification and multi-class classification.

1. Binary Classification

The goal of the binary classification is to separate the ship images from the non-ship images. This requires that we obtain the highest correct classification of a ship present in an image. The performance of correctly classifying a ship present in an image as the number of hidden layer neurons is varied from 1 to 14 neurons(see Figure 31).

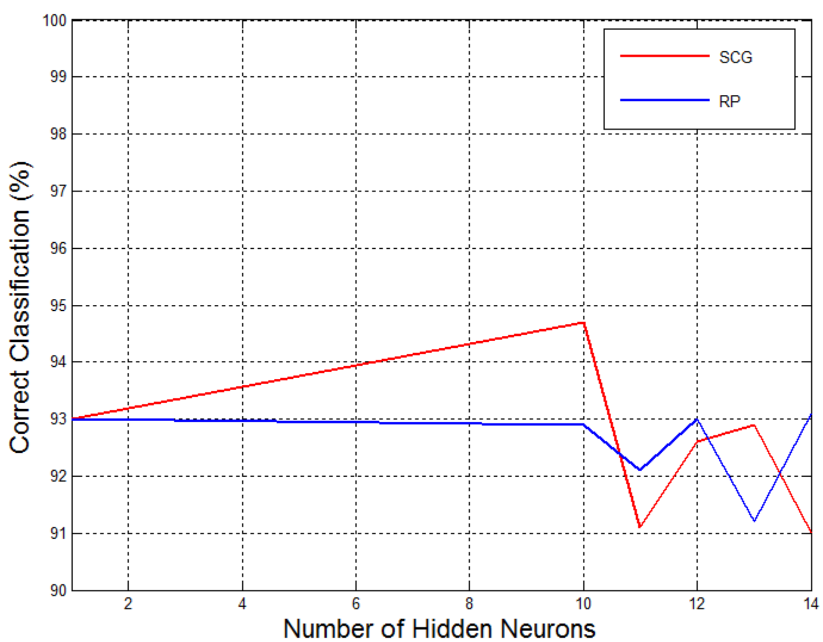


Figure 31. The correct "ship present" classification percentage using geometric and photometric features for training. Classification percentage is a ratio of the number of correctly classified ship images over 1000 trials.

From Figure 31, it can be seen that the neural network was able to separate ship images from non-ship images. There is no clear correlation observed between the number of hidden neurons and classification performance. This phenomenon is likely due to the number of images used for training and testing. If the number of images is increased by an order of magnitude, we expect a more defined correlation in classification performance. Nonetheless, the result of the binary classification application leads to the conclusion that geometric and photometric features are suitable for training a neural network to detect ship images. This is a valuable characteristic of the geometric and photometric features when reducing the amount of images to perform further analysis.

Similar simulations are conducted when the neural networks are trained with the extracted Hough transformation features. The performance of the neural networks when Hough transformation features are used for training is presented in Figure 32.

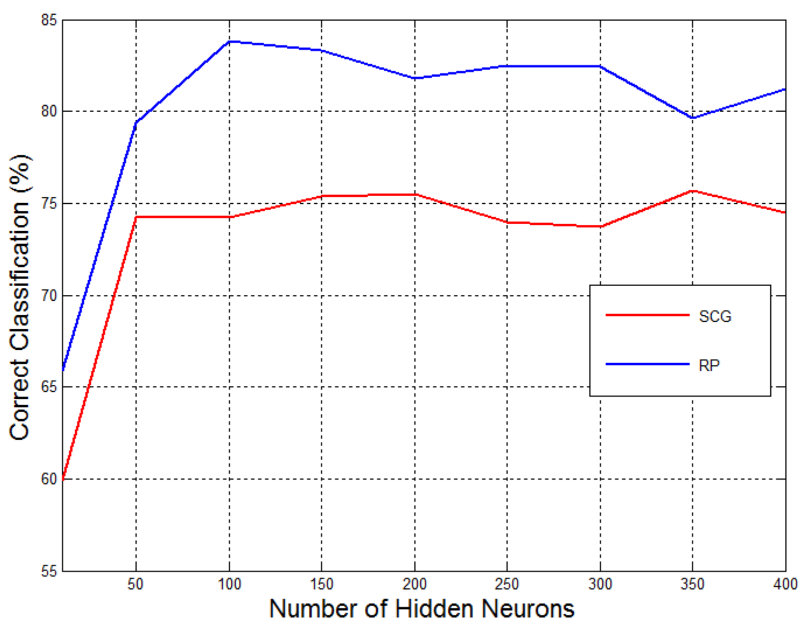


Figure 32. The correct "ship present" classification percentage using Hough transformation features for training. Classification percentage is a ratio of the number of correctly classified ship images over 1000 trials.

From Figure 32, it can be seen that the neural network was able to distinguish ship images from other images in the training set. When the hidden layer is increased to 100 neurons or more, the neural network successfully classified ship images at a high rate. From Figures 31 and 32, we see that geometric, photometric, and Hough transformation features are suitable for training a neural network to detect ship images. The geometric and photometric features yield better classification results; however, the use of the Hough transformation allows for automated feature extraction, fast training times, and successful binary classification rates. The conclusion is that training with Hough transformation features is desired for detecting ship images, but further work is needed to improve its classification performance.

2. Multi-Class Classification

The goal of multi-class classification is to train a neural network to provide a positive classification for one of the four different classes. The targets-of-interest for the purposes of this thesis are images containing warships; therefore, we assess how the neural networks specifically classified the warship images. The highest classification of warship images is desired while reducing the loss of warship images due to misclassification. The performance of correctly classifying the warship images using geometric and photometric features for training is presented in Figure 33.

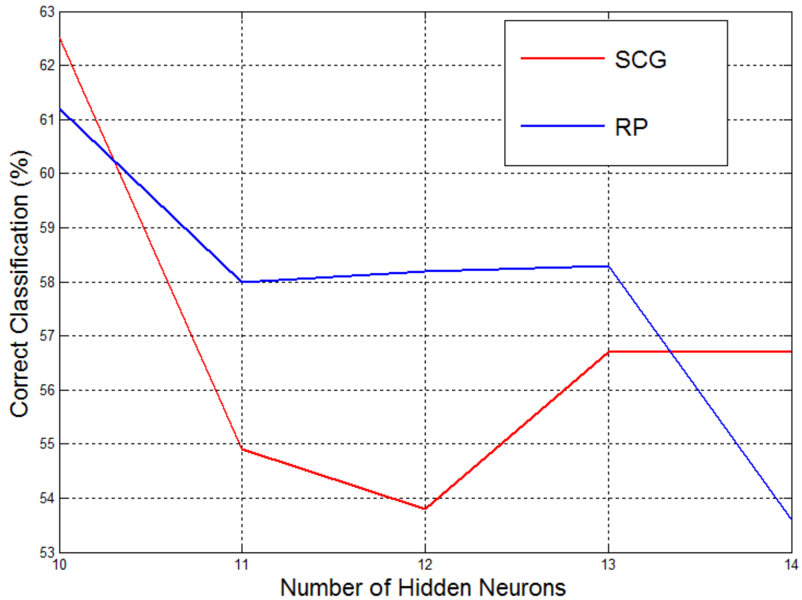


Figure 33. The correct "warship" classification percentage for each learning algorithm and hidden neuron combination. Classification percentage is a ratio of the number of correctly classified warship images over 1000 trials.

From Figure 33, it can be seen that warship classification performance worsens as the hidden layer size is increased. Roughly 40 percent of the warship images are misclassified and lost due to the separation of the image set. The same phenomenon observed in Figure 31 appears again in Figure 33. As explained earlier, a more defined correlation in classification performance is expected as we increase the number of images used for training and testing.

Similar simulations are conducted when the neural networks are trained with the extracted Hough transformation features. The neural network's performance for multi-class classification when Hough transformation features are used for training is presented in Figure 34.

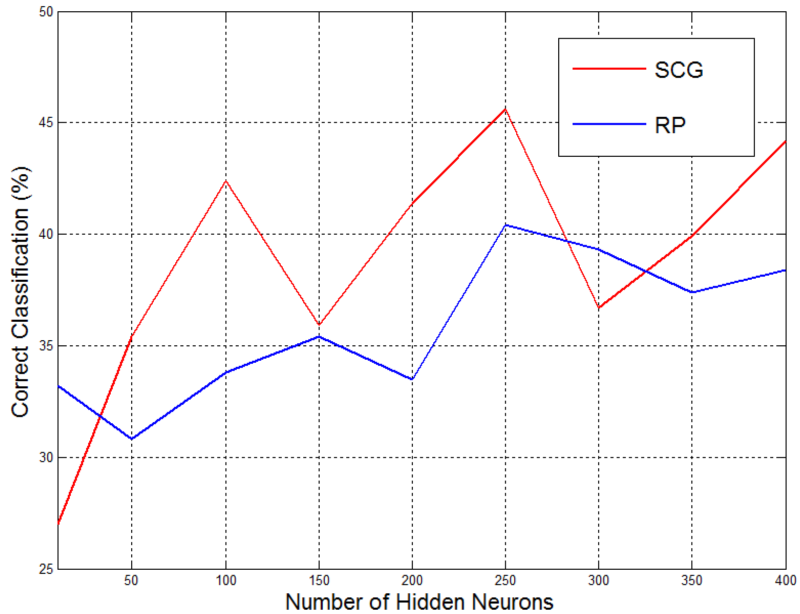


Figure 34. Percentage of correct classification simulations for each hidden neuron combination with the Hough transformation as a training feature.

From Figure 34, it can be seen that warship classification performance using Hough transformation features is unsatisfactory as well. More than half of the warship images are misclassified. An alternative approach to multi-class classification is developed in the next section to explore the potential for achieving better warship classification results.

3. Multi-Level Approach to Multi-Class Classification

Since the performances of the multi-class classification neural networks were poor, an alternative approach, called multi-level classification, was developed in hopes for higher classification rates of individual ship classes. This approach utilizes a series of binary classifications to eliminate one or more ship classes from the image set. The goal is to have only warship images remain by eliminating other image classes with each binary classification test. The approach using multi-level classification is illustrated in Figure 35.

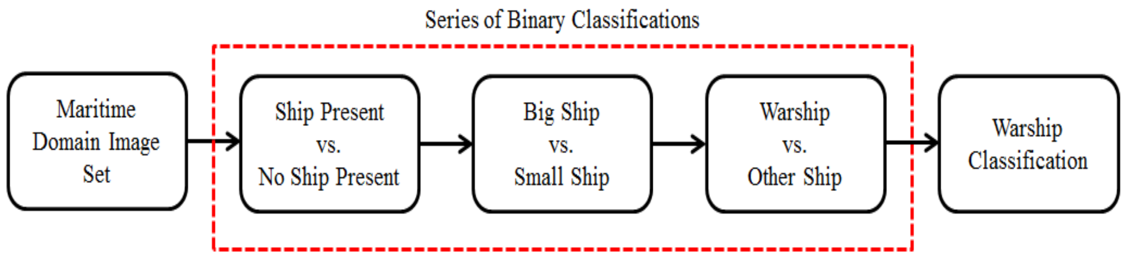


Figure 35. The multi-level classification scheme utilizing a series of three binary classifications. The goal is to eliminate ship classes so that only the warship images remain.

The first binary classification has been previously performed by classifying a ship present or not present from the image set. By removing all of the non-ship images from the image set, the second binary neural network is trained to classify large ships and small ships. This step attempts to reduce the image set to only warship and cargo ship images by eliminating the small boat image class. The performance of the second binary classification is presented in Figure 36.

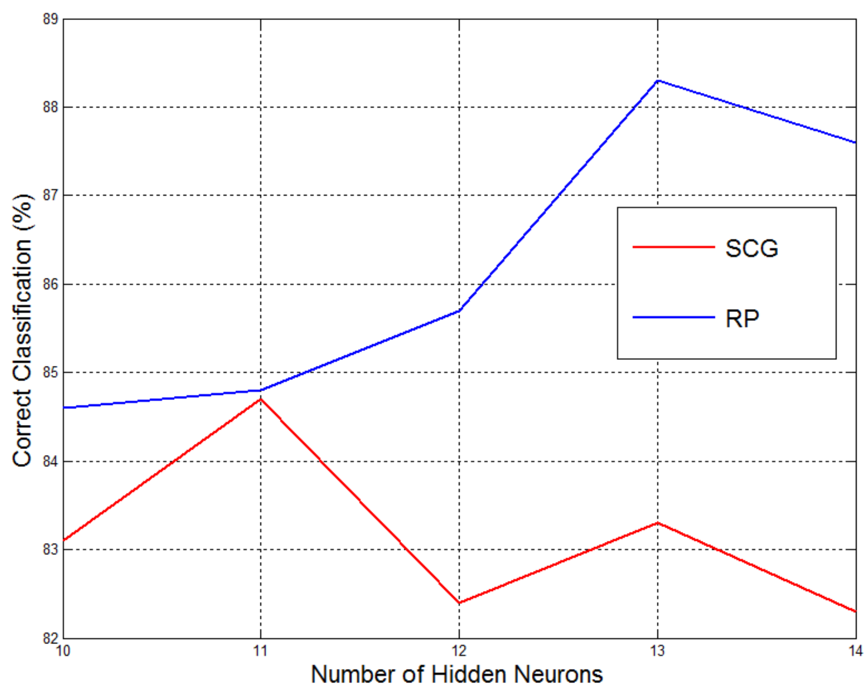


Figure 36. The correct “big ship” classification percentage for the learning algorithms and hidden layer size combination. Classification percentage is a ratio of correctly classified big ship images over 1000 trials.

From Figure 36, it can be seen that the neural network is able to separate large ships from small ships at a rate of at least 82.3 percent and at best 88.3 percent. We use the average of the classification performances as the overall correct classification for this step. The average classification performance when separating big ship images from small ship images is 84.7 percent.

The final binary classification neural network is trained to classify warships from the cargo ship class. This step considers only warship and cargo ship images that are used to train the neural network. The performance of the third binary classification is presented Figure 37.

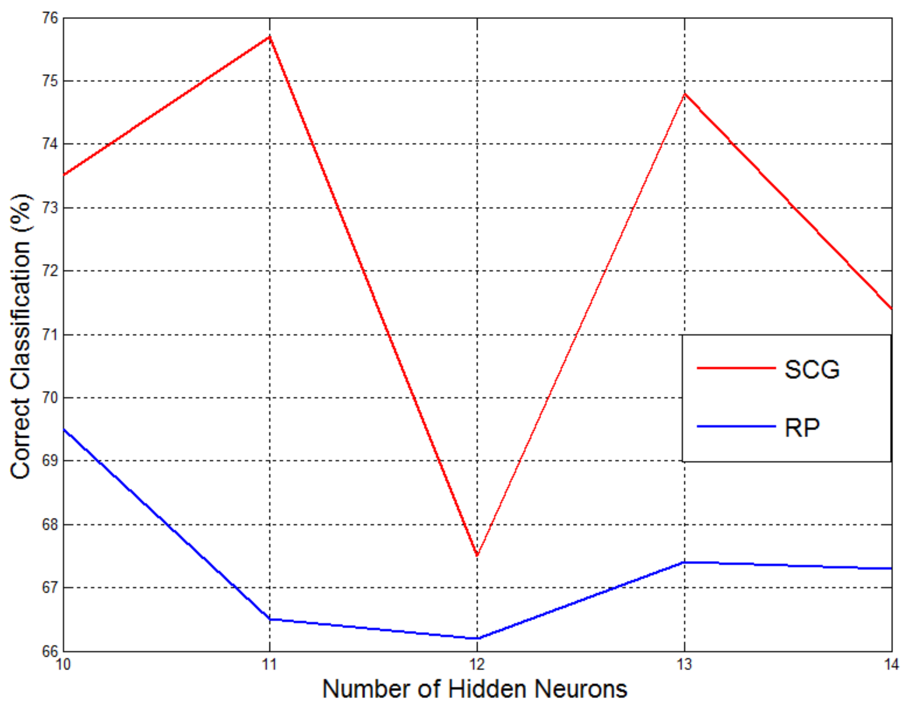


Figure 37. The correct "warship" classification percentage for each learning algorithm and hidden layer size combination. Classification percentage is a ratio of the number of correctly classified warship images over 1000 trials.

From Figure 37, it can be seen that the neural networks separate warship images from cargo ship images at a rate of at least 66.2 percent and at best 75.7 percent. The average classification performance when separating warships images from cargo ship images is 70 percent.

It can be seen from Figures 36 and 37 that the classification performance decreases after each binary classification step. The reason why performance is degraded is most likely due to the reduction in training images after each binary classification. As stated earlier, the reduction in training images may also be the reason for indistinguishable correlation between hidden layer size and classification performance. In the first step, we use all 104 images to train and classify. The training set is then reduced to 54 images for the second binary classification step since the non-ship images are left out. The final binary classification only uses 44 images for training and testing. Although this approach cannot be fairly compared to the multi-class classification implementation, it is fair to state that it is an effective approach to multi-level classification.

The methods outlined in Chapter III were implemented in this chapter describing preprocessing of the maritime domain images and exploring the use of geometric, photometric and Hough transformation features. Implementation of training, testing and classification steps for binary classification to detect a ship present or not present were outlined. Additionally, implementation of training, testing and classification steps for multi-class classification to separate a specific image class from the other maritime domain images was outlined. Finally, a multi-level approach to multi-class classification by using a series of binary classifications is explored.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS

An automated method for improving Maritime Domain Awareness (MDA) was outlined in this thesis by developing neural networks that were trained to identify targets-of-interest. Neural networks were trained to identify targets-of-interest using extracted geometric, photometric, and Hough transformation features from pre-processed images. A binary classification using neural networks was performed to determine whether a ship was present or not present in an image. Additional neural network implementations were developed to perform multi-class and multi-level classification to separate images into four different object classes: warship, cargo ship, small boat, or non-ship.

A. SIGNIFICANT CONTRIBUTIONS

A significant contribution in this thesis is the exploration of neural-network based learning and classification techniques to identify targets-of-interest in the ever-expanding maritime domain in an automated manner. The separation of images that contain targets-of-interest enable a user to allocate more time for further analysis on images that are of actual value.

The study of geometric, photometric, and Hough transformation features as viable input values to neural networks to detect targets-of-interest is also significant. The successful implementation of geometric, photometric, and Hough transformation features to train neural networks to detect ship from non-ship images allows the opportunity for further exploration of this approach.

A third contribution is the development of a multi-level approach to multi-class classification. The neural networks performed significantly better when they were trained to only identify two object classes; however, neural networks performed poorly when they were trained to identify multiple object classes. The multi-level approach was designed to perform a series of binary classifications in order to achieve the same objective as multi-class classification but implemented as a binary tree.

B. RECOMMENDATIONS FOR FUTURE WORK

Several avenues for future work were opened by this thesis. The geometric photometric, and Hough transformation features were effective in training the neural network for classification, but the potential for additional features exists. Future work could expand the scope of features to be extracted to the discrete cosine transformation. Another feature that could be exploited from maritime domain images is the geographic location of the images. Training the neural network to recognize that specific maritime vessels operate in certain geographic locations may help in the classification of such vessels.

The neural network topology investigated in this thesis was limited to one hidden layer. Future work could experiment with different neural network topologies. Adding multiple hidden layers to the neural network may lead to better classification performance.

The images of objects used in this thesis were taken from the overheard, or nearly overhead, aspect. An area of research that can be expanded to develop a more robust classification neural network is to train the neural network using multi-sensor imagery. This method is known as invariance by training [9]. In this approach, a neural network is trained with multiple sensor inputs of the same object. The notion is by introducing enough examples of the same object, the neural network will be capable of generalizing correctly to inputs that it did not experience during the training phase. By expanding the scope of imagery for training, we expect the neural network to provide more robust classification.

The multi-class and multi-level classification schemes did not produce desired performances; however, there is a need to further examine how to improve their classification performance. The ability to reduce the image set to only targets-of-interest would provide a user even more time to analyze images and greatly improve MDA.

APPENDIX A. IMAGE PREPARATION CODE

The MATLAB code used to conduct image preparation is included in Appendix A. Comments are in green text.

```
% Load the original, unaltered image
I = imread('PE46.jpg');

% Convert original image to grayscale
I2 = rgb2gray(I);

% "my small number" created to avoid dividing by zero circumstances
my_small_number = 1;
I2(I2==0) = my_small_number;

% Rescale values to be between [0,1]
I2 = double(I2);
B = size(I2);
B = 255* ones(B);
intensity = I2./B;

% Histogram of grayscale for object selection
imhist(intensity)

% Use histogram to segment image
hist = imhist(intensity);

[pks, locs] = findpeaks(hist);   % Find peaks
d = abs(locs(2,1)-locs(5,1));    % Calculate the distance between peaks
d = (d/2) + locs(2,1);           % Locate midpoint of peaks (threshold
value)
d = d/255;                       % Normalize threshold value

BW = im2bw(intensity, d);        % Convert image to black and white
using the threshold for segmentation
imshow(BW)

% Erode noise from image by removing everything less than specified
disk size
seD = strel('disk',16);
BWfinal = imerode(BWdfill,seD);
BWfinal = imerode(BWdfill,seD);
figure, imshow(BWfinal), title('segmented image');

% Dilate object in image ti fill in holes
BWdfill = imfill(BW,26,'holes');
figure, imshow(BWdfill);
title('binary image with filled holes');
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. FEATURE EXTRACTION CODE

The MATLAB code used to conduct feature extraction is included in Appendix B. Comments are in green text.

```
% Perimeter (P)
Perimeter = bwperim(BWfinal);
imshow(Perimeter);
P = find(Perimeter);
P = length(P)

% Area (A)
A = bwarea(BWfinal)

% Complexity (C)
C = (P)/(2*sqrt(pi*A))

% Spreading (S)
[I,J] = find(BWfinal);  % Locate the indices of the object
cov_matrix = cov(I,J);  % Compute the covariance matrix of the object
eig_values = eig(cov_matrix);  % Compute the eigenvalues of the
covariance matrix

lamda1 = max(eig_values);  % Compute the maximum eigenvalue
lamda2 = min(eig_values);  % Compute the minimum eigenvalue

S = (100*lamda2)/(lamda1+lamda2)

% Object standard deviation (OSd) in dB
B = size(I2);
B = 255* ones(B);
intensity = I2./B;
obj_intensity = BWfinal.*intensity;
obj_intensity = nonzeros(obj_intensity);
obj_int_std = std(obj_intensity);  % Standard deviation of object
intensity values

OSd = 10*log10(1/obj_int_std)  % Convert value into dB

% Background standard deviation (BSd) in dB
background = BWfinal - 1;
background = abs(background);
back_int = background.*intensity;
back_int = nonzeros(back_int);
BSd = std(back_int);  % Standard deviation of object intensity values

BSd = 10*log10(1/BSd) % Convert value into dB

% Max contrast (ConMax)
% Difference (in dB) between the background mean value and the lowest
value inside the object.
```

```
background_mean = mean(back_int);
object_min = min(obj_intensity);
ConMax = abs(background_mean - object_min);

ConMax = 10*log10(1/ConMax)  % Convert value into dB

% Mean Contrast (ConMe)
% Difference (in dB) between the background mean value and the object
mean
% value
object_mean = mean(obj_intensity);
ConMe = abs(background_mean - object_mean);

ConMe = 10*log10(1/ConMe)  % Convert value into dB

% Max Gradient (GMax)
% Maximum value (in dB) of border gradient
[FX, FY] = gradient(intensity);
perim_grad_x = FX.*Perimeter;
perim_grad_y = FY.*Perimeter;
perim_grad1 = [perim_grad_x perim_grad_y];
perim_grad_abs1 = abs(perim_grad1);
perim_grad_abs1 = nonzeros(perim_grad_abs1);
GMax = max(perim_grad_abs1);

Gmax = 10*log10(1/GMax)  % Convert value into dB

% Mean Gradient (GMe)
% Mean border gradient
GMe1 = mean(perim_grad_abs);

GMe1 = 10*log10(1/GMe1)  % Convert value into dB

% Gradient Standard Deviation (GSd)
% Standard deviation (in dB) of the border gradient values
GSd1 = std(perim_grad_abs1);

GSd1 = 10*log10(1/GSd1) % Convert value into dB
```

APPENDIX C. HOUGH TRANSFORMATION CODE

The MATLAB code used to conduct image preparation, conduct Hough transformation, and extract features from the Hough space is included in Appendix C. Comments are in green text.

```
% Load Image
I = imread('PE46.jpg');

% Resize to 256x256
I = imresize(I, [255 255]);

% Convert to Grayscale
I = rgb2gray(I);

% Edge detection
I = edge(I,'sobel');

% Convert to 'double'
I = double(I);

figure(1)
imshow(I)

% Hough Transform
[H,T,R] = hough(I);

% Plot of Hough Space
figure(2)
imshow(imadjust(mat2gray(H)))
colormap(hot)

% Determine peak vector in Hough Space and plot
[H_max,r] = max(H);
H_max = H_max/max(H_max);
r = r/max(r);

% Input vector for image (Peak vector and Rho Vector)
x = [H_max r];
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. NEURAL NETWORK TRAINING AND TESTING CODE

The MATLAB code used to conduct neural network training and testing is included in Appendix D. Comments are in green text.

```
% Feature values extracted from each image
x = x;

% Object class target values
t = t;

% Two-layer (i.e. one-hidden-layer) feed forward neural networks can
learn
% any input-output relationship given enough neurons in the hidden
layer.
% Layers which are not output layers are called hidden layers.
% In general, more difficult problems require more neurons, and perhaps
% more layers. Simpler problems require fewer neurons.

% Set the size of the hidden layer. "Number" indicates the number of
neurons in the hidden layer
hiddenLayerSize = Number;

rng(0);   % Restore the generator settings
net = patternnet(hiddenLayerSize);  % Activate the Neural Network
Toolbox
view(net)

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 0.99;  % Percentage of the inputs used for
training
net.divideParam.valRatio = 0/100;  % Percentage of the inputs  used for
validation
net.divideParam.testRatio = 0.01;  % Percentage of the inputs used for
testing

% "for" loop for running multiple trials. The results of each
simulation will be collected.
 plt = 0;
 Ntrials=1;  % Set number of trials to be executed
 for i = 1:Ntrials
   s{i} = rng('shuffle'); % Shuffle the inputs used for training
   net  = configure(net,x,t);
   net.trainFcn='trainscg';  % Learning algorthm used for training
   net.trainParam.epochs=100000; % Maximum number of iterations to
finish training
```

```
   [net tr y e] = train(net,x,t); % Train neural network with inputs
and their defined target values
            netIW0{i} = net.IW;     % Input weights
            netb0{i}  = net.b;      % Network bias
            netLW0{i} = net.LW;     % Layer weights
         tstind = tr.testInd;
         ytst{i} = y(:,tstind);
         ttst{i} = t(:,tstind);
         epoch{i} = length(tr.epoch);  % Collect iterations for
convergence in each simulation
 end

plotconfusion(ttst,ytst)  % Plot confusion matrix
title('TEST SET CONFUSION MATRIX')
hold off

epoch_lengths = cellfun(@mean, epoch); % Collect the lengths of
iterations for each simulation
mean_epoch = mean(epoch_lengths)  % Calculate the average number of
iterations to converge
```

LIST OF REFERENCES

[1] *Maritime Security Policy*, National Security Presidential Directive NSPD-41, George W. Bush Administration, Washington, DC, 2004, pp. 1–9.

[2] U.S. Department of Homeland Security. (2005, Oct.). National plan to achieve maritime domain awareness. [Online]. Available: https://www.dhs.gov./national-plan-achieve-maritime-domain-awareness

[3] C.K. Chan and M.B. Sandler, "A complete shape recognition system using the Hough transform and neural network," Proc. Intl. Conf. on Image Processing and its Applications, University of London, 1992.

[4] F. Del Frate, A. Petrocchi, J. Lichtenegger, and G. Calabresi, "Neural networks for oil spill detection using ERS-SAR data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, no. 5, Sept. 2000.

[5] O. M. Soysal, J. Chen, and H. Schneider, "An image processing tool for efficient feature extraction in computer-aided detection systems," In *Proc. IEEE Conf. on GrC*, Silicon Valley, CA, 2010.

[6] Gaurav Kumar and Pradeep Kumar Bhatia, "A detailed review of feature extraction in image processing systems," In *Proc. 4th Intl. Conf. on Advanced Computing & Commun. Technol.*, Rohtak, India, 2014, pp. 5-12.

[7] X. Yun and K. Latt, "Mobile robot localization using the Hough transform and neural networks," In *Proc. ISIC/CIRA/ISAS Joint Conf.*, Gaithersburg, MD, 1998, pp. 393-400.

[8] K. Huang, J. You, K. Chen, H. Lai, and A. Dong, "Hough transform neural networks for seismic pattern detection," In *Proc. Intl. Joint Conf. on Neural Networks*, Vancouver, BC, Canada, 2006, pp. 2453–2458.

[9] Simon Haykin, *Neural Networks: A Comprehensive Foundation,* 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999, pp. 1-155.

[10] R.P. Lippmann, "An Introduction to computing with neural nets," *IEEE ASSP Magazine.*, vol. 4, no. 2, pp. 4–22, April 1987.

[11] Mathworks. (n.d.) *trainrp* [Online]. Available: http://www.mathworks.com/help/nnet/ref/trainrp.html

[12] M.F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning," Tech. Rep. DAIMI PB-339. Comp. Sci. Dept., Univ. Aarhus, Denmark., Nov. 1990.

[13] Mathworks. (n.d.) *rgb2gray* [Online]. Available: http://www.mathworks.com/help/matlab/ref/rgb2gray.html

[14] Mathworks. (n.d.) *imhist* [Online]. Available: http://www.mathworks.com/help/images/ref/imhist.html

[15] Mathworks. (n.d.) *findpeaks* [Online]. Available: http://www.mathworks.com/help/signal/ref/findpeaks.html

[16] Mathworks. (n.d.) *im2bw* [Online]. Available: http://www.mathworks.com/help/images/ref/im2bw.html

[17] Mathworks. (n.d.) *imfill* [Online]. Available: http://www.mathworks.com/help/images/ref/imfill.html

[18] Mathworks. (n.d.) *imerode* [Online]. Available: http://www.mathworks.com/help/images/ref/imerode.html

[19] Mathworks. (n.d.) *bwarea* [Online]. Available: http://www.mathworks.com/help/images/ref/bwarea.html

[20] Mathworks. (n.d.) *bwperim* [Online]. Available: http://www.mathworks.com/help/images/ref/bwperim.html

[21] Mathworks. (n.d.) *cov* [Online]. Available: http://www.mathworks.com/help/matlab/ref/cov.html

[22] Mathworks. (n.d.) *eig* [Online]. Available: http://www.mathworks.com/help/matlab/ref/eig.html

[23] Mathworks. (n.d.) *std* [Online]. Available: http://www.mathworks.com/help/matlab/ref/std.html

[24] Mathworks. (n.d.) *mean* [Online]. Available: http://www.mathworks.com/help/matlab/ref/mean.html

[25] Mathworks. (n.d.) *min* [Online]. Available: http://www.mathworks.com/help/matlab/ref/min.html

[26] Mathworks. (n.d.) *gradient* [Online]. Available: http://www.mathworks.com/help/matlab/ref/gradient.html

[27] Mathworks. (n.d.) *max* [Online]. Available: http://www.mathworks.com/help/matlab/ref/max.html

[28] Mathworks. (n.d.) *imresize* [Online]. Available: http://www.mathworks.com/help/images/ref/imresize.html

[29] Mathworks. (n.d.) *edge* [Online]. Available: http://www.mathworks.com/help/images/ref/edge.html

[30] Mathworks. (n.d.) *hough* [Online]. Available: http://www.mathworks.com/help/images/ref/hough.html

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California